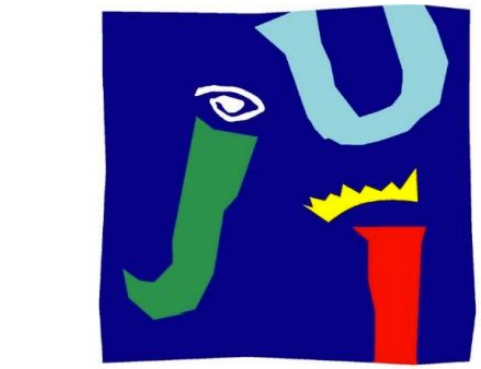


UNIVERSITAT JAUME·I

University of Salford MANCHESTER

Jaume I University

in collaboration with the University of Salford

Degree on Design and Development of Videogames

Final Degree Project Report

Author: Alejandro Gutiérrez Rodríguez

Tutor (UJI): María del Mar Marcos López

Tutors (UoS): Michael Lockwood / Juan Hiriart

Summary

This project aimed to mimic a real life videogame studio work experience by gathering the different departments that conform a development team in order to design and develop a game that excels both for its aesthetics as for its mechanics.

To accomplish this task the team decided to follow the footsteps of some of the great geniuses in the history of narrative as J. R. R. Tolkien or George R. R. Martin and develop *Relinquish*, a fantasy medieval themed action / adventure game. To stand out over the rest of games of this genre, *Relinquish* introduces a new level down system that makes the game a challenge for the players.

The aesthetics of the game were made by the art department in which a was part of, for this reason, this report will be divided in both the whole project put together by the team and the artistic part made by myself, which consisted of three dimensional modelling and texturing along with a limited programming part.

Key Words

Work team, medieval narrative, aesthetics, three dimensional modelling

Index

Figure Index

Chapter 1:
Technical Proposal

Introduction

In videogames history, the action / adventure genre has been heavily exploited because it fits great with their interactivity. Moreover, medieval style has been one of the most acclaimed, both in the world of videogames as in film industry and television. Proof of this are works as *The Lord of the Rings*[1] or *Game of Thrones*[2], among many others, which pose a made up world that results very similar in the most aspects to the european Middle Age.

The main motivation of the group is to join this great world of audiovisual works, taking advantage of its existence and main features as a starting point, but including our own touch of creativity to create something that, despite of belonging to a widely exploited genre is differentiable from other works.

The group's proposal is, therefore, the development of an action-adventure video game of medieval style based on the protagonist's attempt to escape from the prison in which he is being held. To achieve this he must fight the guards who watch the catacombs, while he is forced to solve puzzles to advance. All this along with a revolutionary level down system: a concept that will differentiate this game from others, since as the game progresses enemies do not become more powerful, but, due to a curse, the character controlled by the player will weaken, so that the difficulty of the game will be greater and will pose increasingly complex challenges for the player.

For my personal proposal, as a stage asset designer my goal is to model and texturize as many objects as possible that turn the player experience into an immersion in medieval times. I will try to create elements with the highest possible level of detail in a realistic style. As for the programming section in which I will take part in the project, my mission is, on the one hand to develop a camera in the third person able to follow the player, rotate around him and solve the problems that this would cause, such as crossing through walls or turning upside down when rotating vertically; On the other hand, I will develop an AI system for a special character who will act as the final boss. In addition to its programming, I will also take care of its design, modeling, textures and animation.

Related Subjects:

- VJ 1208: Programming II
- VJ 1216: 3D Design
- VJ 1226: Character Design and Animation
- VJ 1227: Game Engines
- VJ 1231: Artificial Intelligence

Objectives:

- **Objectives (as a group):**
  - Create a game based in the action / adventure genre with medieval style that differentiates itself from the rest by its unique qualities.
  - Perform a job close to a commercial level.
  - Experience the group work the same way as in a professional level.

- **Objectives (on a personal level):**
  - Design, model and texturise quality models for the game scenario.
  - Program a profesional third person camera controller.
  - Design and program the AI of a character who acts as final boss of the level.

Task Planning (on a personal level):

1. Technical proposal (3 hours):
   - Writing of the technical proposal (3 hours)
2. Artistic Design (20 hours):
   - Character design (10 hours)
   - Environment assets design (10 hours)
3. Modeling (120 hours):
   - Character modeling (60 hours)
   - Environment assets modelling (60 hours)
4. Texturing (50 hours):
   - Character texturized (20 hours)
   - Environment assets texturized (30 hours)
5. Animation (20 hours):
   - Character animation (20 hours)
6. Programming (35 hours):
   - Programming the third person camera controller (5 hours)
   - Programming the movement and animation of the final boss (10 hours)
   - Programming the artificial intelligence of the final boss (20 hours)
7. Documentation (20 hours):
   - Preparation of the final report of the project (15 hours)
   - Elaboration of the final personal memory (5 hours)
   - Elaboration of the presentation before the court (5 hours)

Below is a chart showing the distribution of hours per week:

Task	Hours	1	2	3	4	5	6	7	8	9	10	11	12
Technical proposal	3	3											
Artistic Design	20	15	5										
Modeling	120		20	20	20	20	20	20					
Texturing	50							10	20	20			
Animation	20									10	10		
Programming	35					10					15	10	
Documentation	20											5	15

Expected results:

Expected results (as a group):

- Achieve the development of a brief game of high quality at a visual and mechanical level.
- Learn the workings of a real working group of the video game industry.

Expected results (on a personal level):

- Develop a set of models with their respective professional quality and reusable textures.
- Develop the model of a character and his high quality textures and animations.
- Program an artificial intelligence and a third person camera controller.

Chapter 2:
Related Work

Game references

As reference for our game we considered the following games and series:

- ***The legend of Zelda*[3] series.**

The Legend of Zelda is a fantasy action / adventure video game series developed and published by Nintendo. The series' gameplay incorporates elements of action, adventure and puzzle-solving games.

The entire series as general we took inspiration from, as it was very similar to our game concept, set in medieval times with players taking on the role of a young swordsman who is guided by an old man and sets off on his journey towards the kingdom's castle. The environments *The Legend of Zelda* also uses such as the villages, castles and big fields is all environments we wanted to explore in our game. The visual style, particularly the one used in *The Legend of Zelda: Breath of the Wild* we wanted to produce something similar for our game.

- ***The Elder scrolls*[4] series.**

The Elder Scrolls is a series of action role playing open world fantasy video games developed by Bethesda Game Studios. The series is known for its elaborate and richly detailed open worlds and its focus on free-form gameplay.

The Elder Scrolls is also a perfect reference for our game because is basically a massive version of the game we were trying to develop. The biggest difference comes with the huge open world because we do not had enough time to design and produce it so it is possible to say that our game's world is pretty similar to a dungeon of this videogame series. The rest of the game is very similar: fantasy medieval style, realistic characters and environments and puzzle solving.

- ***MediEvil*[5] series**

MediEvil is a series of three action / adventure hack and slash video games developed by SCE Cambridge Studio The series revolves around an undead charlatan knight in his attempt to restore peace.

MediEvil is one of the series that set the medieval hack and slash videogame style, so we wanted to pay tribute to it focussing on the simple and fast combat system that made the game so popular at a point. Something interesting about this reference is that at the final stage of our game's development we decided to change the boss's appearance to make him look similar to *MediEvil's* main character.

- ***World of Warcraft***[6]

World of Warcraft (WoW) is a massively multiplayer online role-playing game (MMORPG) released by Blizzard Entertainment.

WoW is a game that includes a huge amount of caves and caverns along the huge map. This is the point of having this game as reference, trying to learn as much as possible about how this game converts a simple cave in an epic puzzle full of enemies.

- ***Diablo*[7] series**

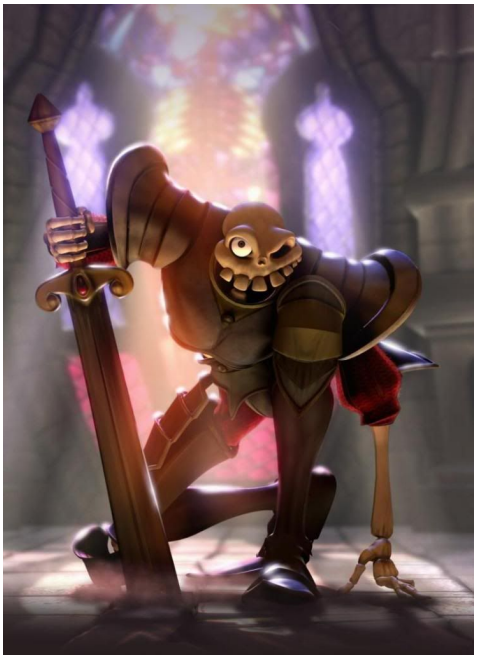
Diablo is an action role-playing hack and slash dungeon crawler video game series developed by Blizzard Entertainment.

The hack and slash part of this game is what interested us the most along with the realistic graphic style of the last game of the series, *Diablo III*. A big part of the success of this series is the dynamism of the fast combat system joined to the design and the huge world.

References for the individual work

When it come to the environmental assets I wanted to create everything in a very realistic way. For this reason I had to took AAA games as references. As mentioned before, in our game we wanted to recreate the realism of games as *The Elder Scrolls V: Skyrim*, *The Legend of Zelda: Breath of the Wild* or *Diablo III*, so to make the objects for the game I used mainly Skyrim, as in the pictures below.

For the second part of my job, the final boss, I decided to take the main character of the *MediEvil* series along with the character in the film *Ghost Rider,* a human swordsman with a skull as head. At first I created the character with the naked torso but at the end I decided that making him wear an armour would make much more sense, so the final versión is really similar to the *MediEvil* character but in a more realistic modelling style. To model the skull took reference of a picture of an actual skull.



Previous work

As the group was founded for the realization of the project there is not a previous work done together, so in this subsection I'm going to briefly talk about some of my recent 3D models (because my role in the group was 3d modeller), showing some pictures.

This mood board below corresponds to a project for a subject calle 3D Design is which we had to create a 3D model of an object, in my case a Formula One car, apply textures to it and put it in a simple environment. At a glance you can see the level of detail is quite high, specially focussing in the bottom left picture, where we can see that the brakes are modelled too even though it is pretty hard to see them when the model is assembled.

Another recent project was the development of a stealth genre videogame for a subject called Artificial Intelligence. In this case the artistic part was way more basic than in the previous example, only having to model simple assets and create a basic classroom environment as shown in the pictures below.

Chapter 3:
Design

Statement:

Relinquish is a medieval third person action RPG that aims to immerse players into its fantasy world through interesting characters and engaging combat. Our game is aimed at ages twelve and up and an audience who enjoys fantasy and medieval settings.

Through engaging combat and an intriguing narrative, we are able to create a game that appeals to both hardcore and casual gamers. The game has the potential to be released across both console and PC enabling a wider user base to engage with our game.

Our game concept was to either follow an adventure game or handle a more serious topic such as children of war however after discussing idea's we landed on the idea of the adventure game and setting it in medieval times. The player would take on the role of the protagonist named Zephyr which begins the game in a prison cell in the castle dungeon with a case of amnesia. An old man in the neighbouring cell would then greet the player and give them their objectives to begin to progress through the game. The goal of the game would be to restore the player's character's memory to discover who they are and what happened to the kingdom.

Our presented concept is that of a medieval theme basing it around traditional lore of kingdoms, kings and sorcery. We also take on the concept of playing the anti-hero like in games such as – *God of War*[8]*, Prototype*[9] and *Shadow of the colossus*[10]*.*

Our game also takes on a lot of traditional fantasy themes such as containing fantasy creatures like goblins and being able to consume magic potions to restore health. A big influence and of the same topic was the BBC show *Merlin*[11] along with the whole mythos of King Arthur[12]. Games such as *The legend of Zelda* and *The Witcher*[13] were a big influence and follow a similar theme.

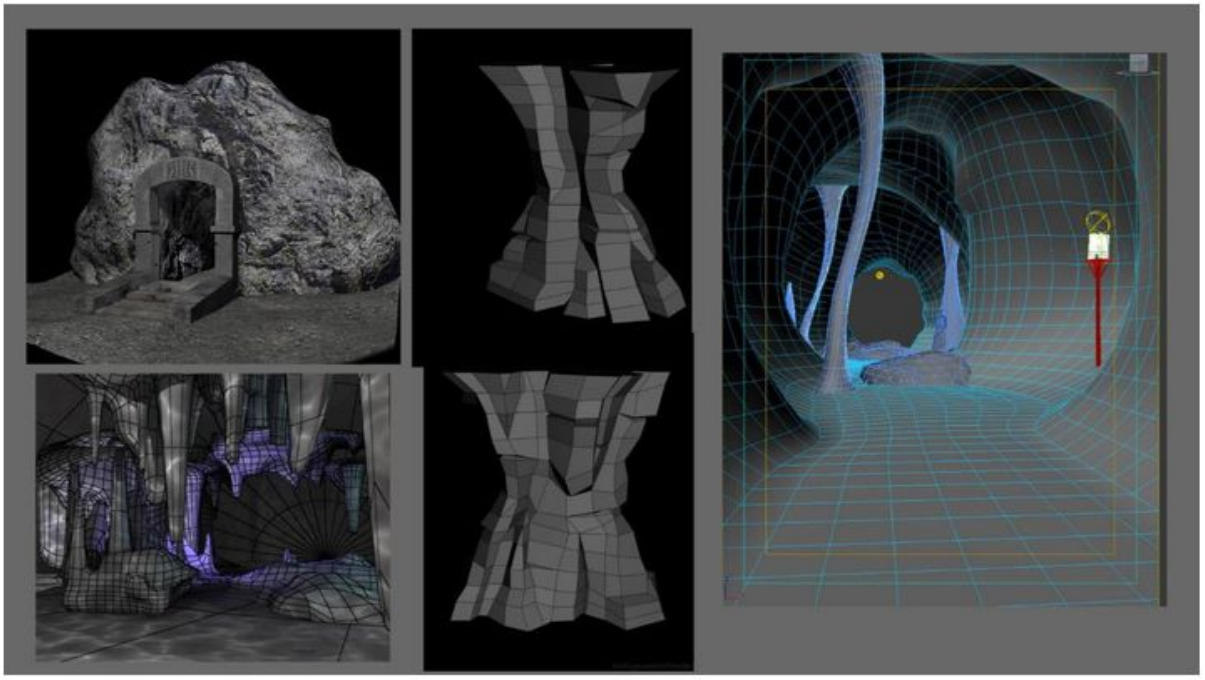
We aimed to stay true to this concept in our aesthetics and nature of storytelling. When fleshing out this world we wanted it to take on the creative stride of fantasy creatures and have interesting weapons with lore behind them to intrigue the player and immerse them further into this mythos. Atmosphere is also something we hoped to capture the dimly lit dungeons, bright green grass and wooden old villages.

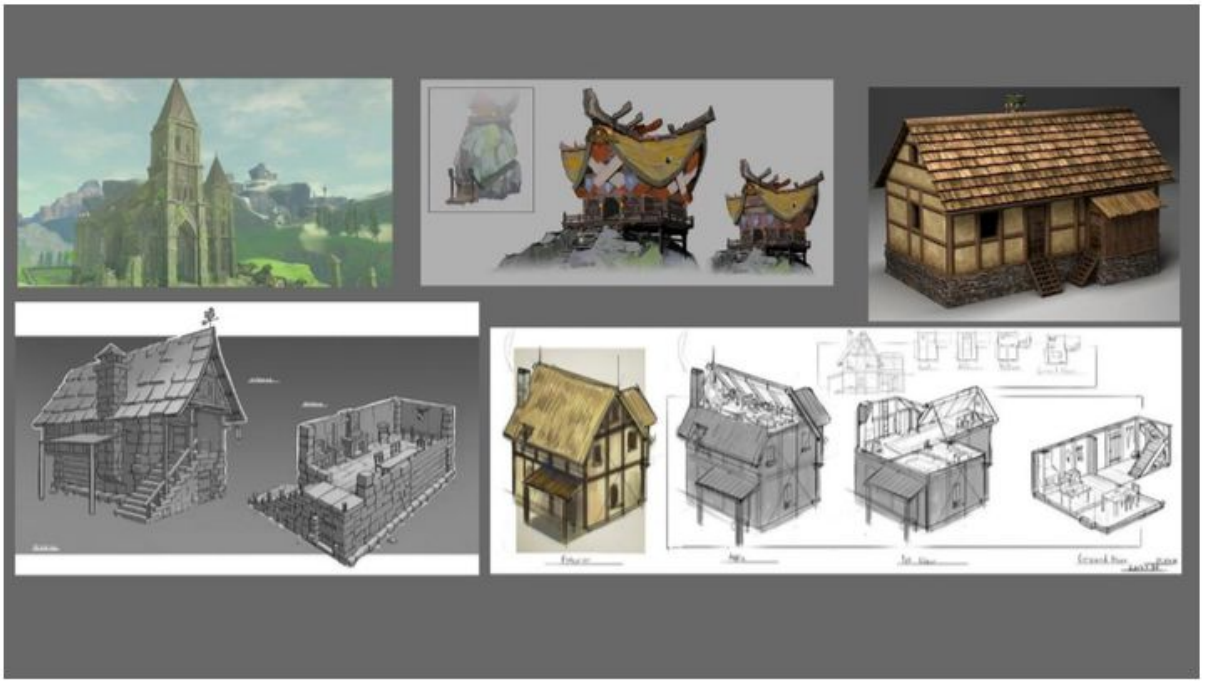
Aesthetics:

To begin to visualise a style and overall theme for our game we created some mood boards for characters and environment.

Character mood board:

Environment mood boards:





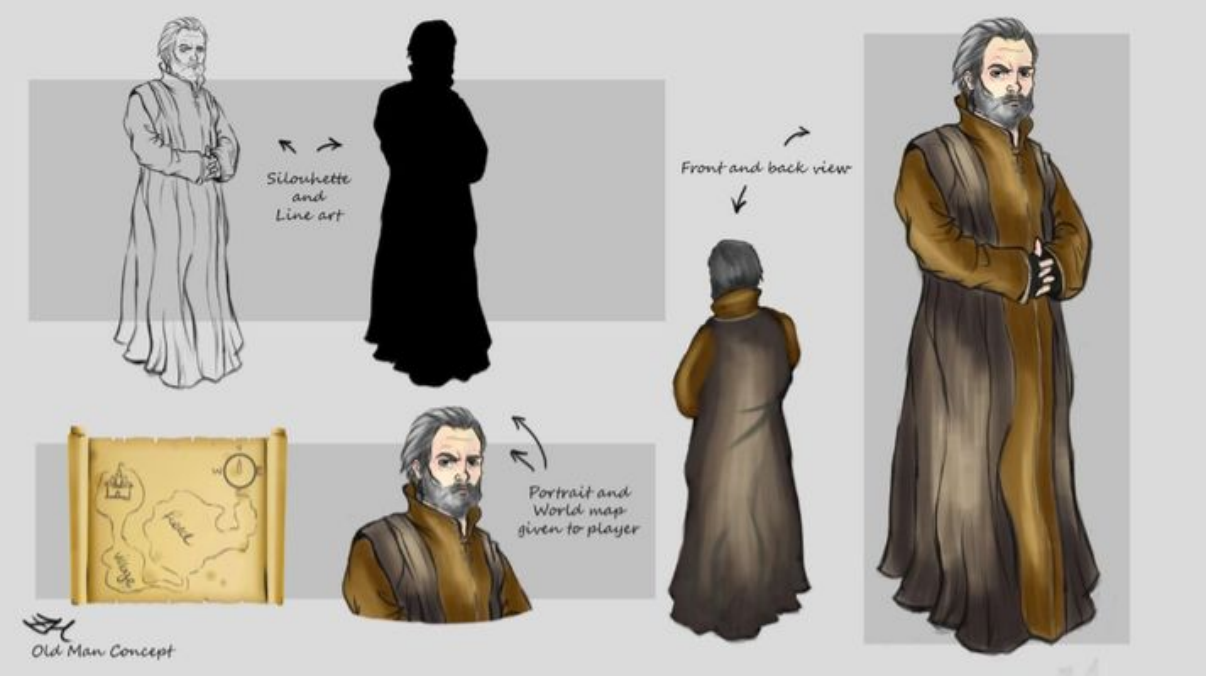
Character Design:

The character art was all designed by one artist to produce a consistent visual style in these designs and when transferring these over to 3D models. The first character to be conceptualised was the protagonist.

The image shown to the right is the first design sporting a more heroic approach to the overall design. The design was very much inspired by a traditional rendition of “King Arthur” with the blonde hair and red cape.

The image above shows the character’s design overhaul after discussing the narrative. We aimed to put a twist in the narrative to make the overall story more interesting and give the characters more depth, this twist however made the protagonist to secretly turn out to be the villain. With this in mind his design was given a darker tone using a darker colour palette but also not trying to give the twist away. The design matches a perfect balance between a heroic protagonist and villainess swordsman.

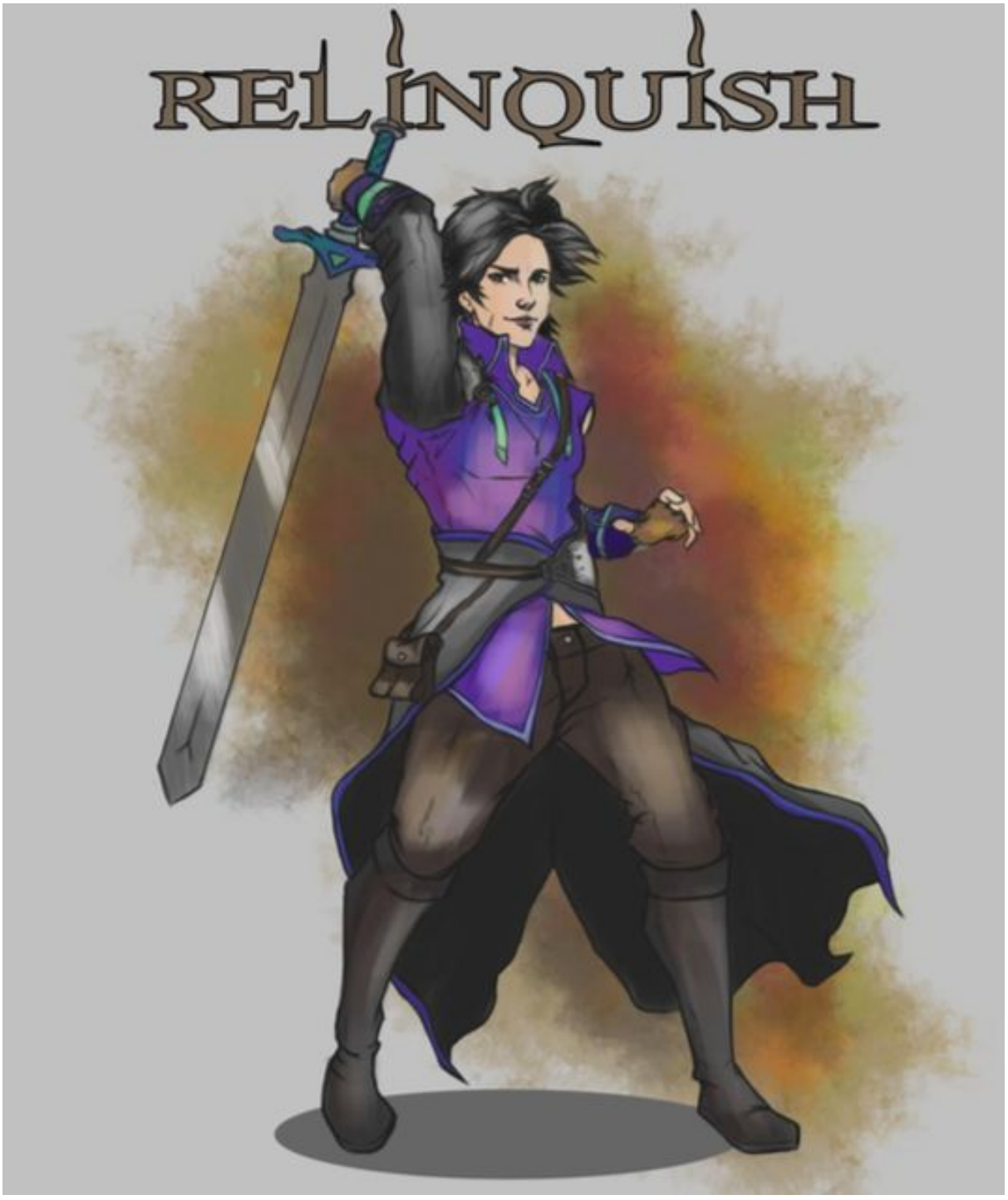
An expression sheet of the protagonist was also created to get a feel for the personality of the character more and how to potentially animate his face at a later stage on the model.



The image above shows the character sheet for the second character design which would be the old man who would give you objectives throughout the game. You'd first encounter him right at the beginning of the game as he is in the neighbouring cell to you.

The character sheets follow a similar style to the protagonist one although in the bottom left corner of the image is a rough example of a map of the kingdom you'd explore once you escaped from the dungeon that the old man would give to the player.

The following cover art was produced to summarise our game and give the game a poster for marketing purposes. The image is simplistic focusing on the protagonist with flames similar to the ones used in the title screen along with the title at the top of the piece.

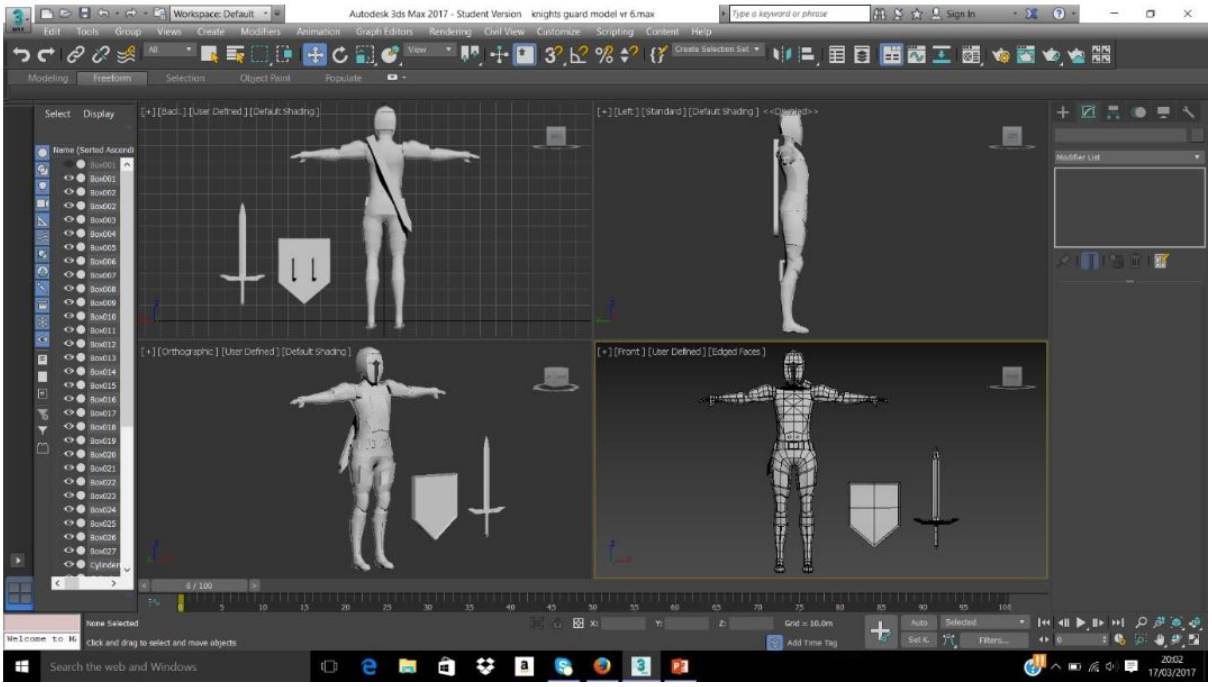


Character Models

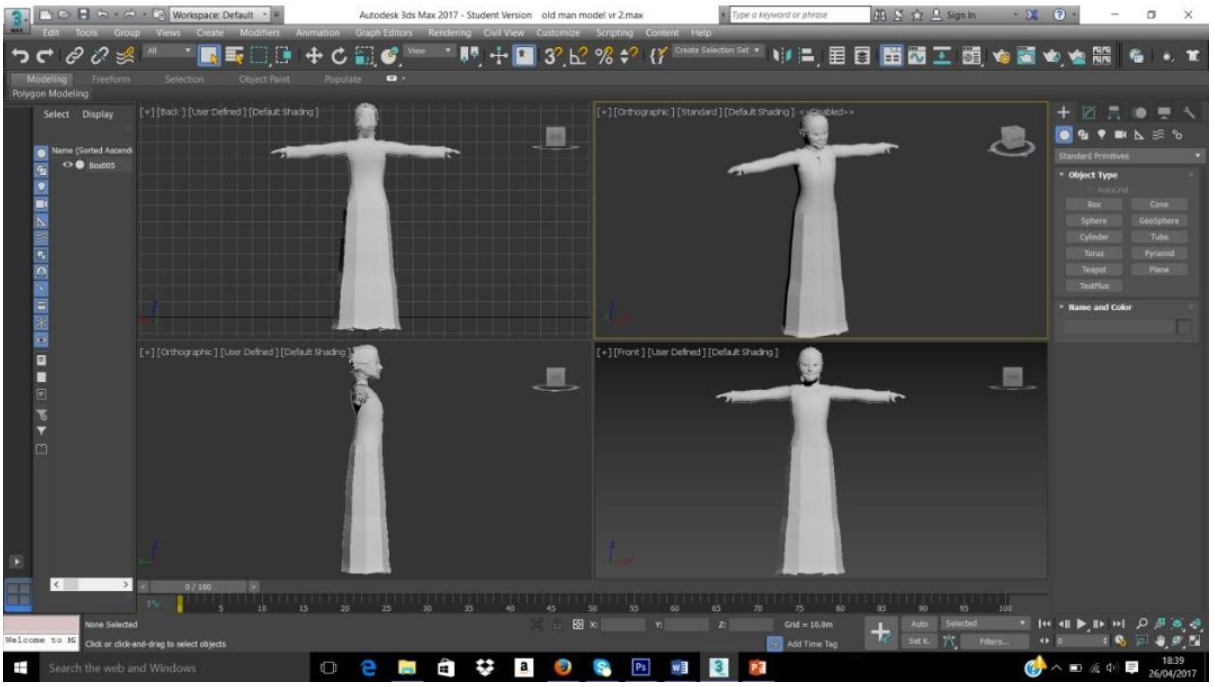
Our character artist also created the character models for our game along with texturing, rigging and animating would be passed over to another member of the team.

In the above image is the grey box of our protagonist character that the player will control. This was included in the version of the game that was used for external play testing and was well received amongst our audience as well as the other character models.

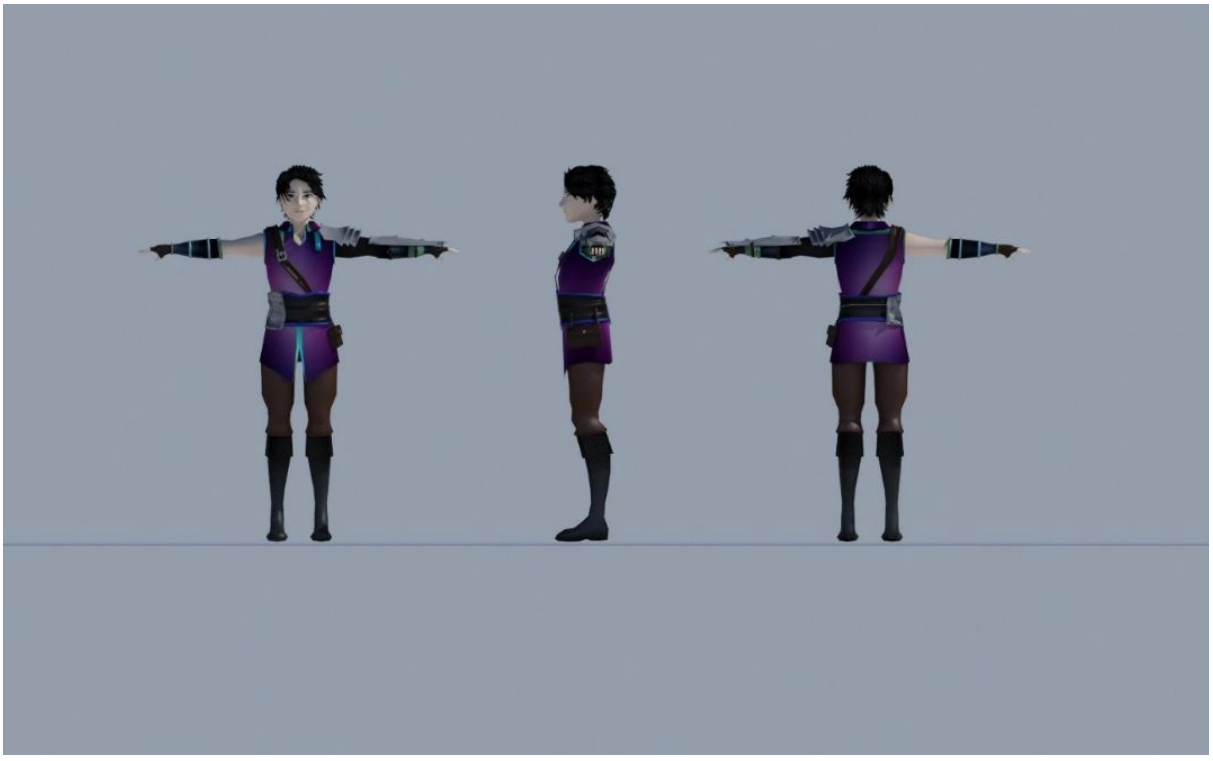
In this image is the grey box guard that will attack the player in the game. Beside him is a modelled shield and sword that he will wield to attack the player.



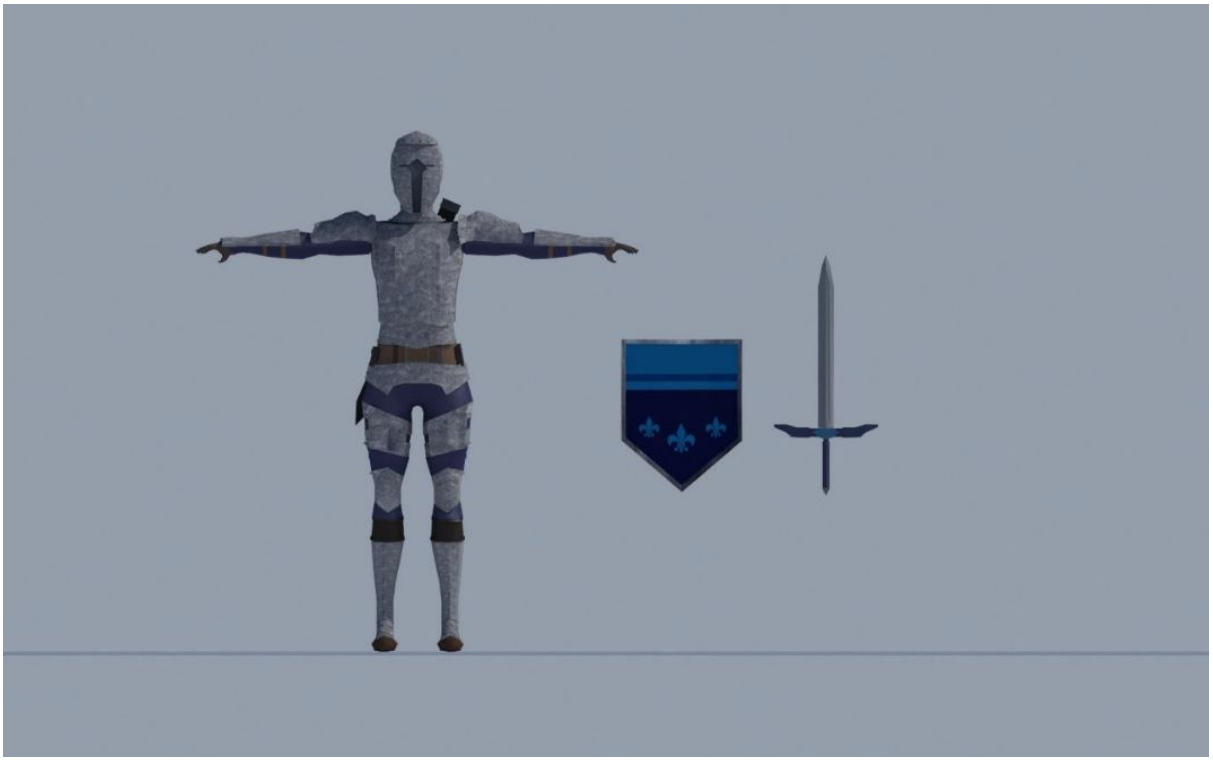
Finally, above is the grey box of the old man character who interacts with the player and gives them their objectives. His model was more basic compared to the previous two due to his simplified clothing design.



In the following image is a high quality render of the final, textured protagonist model showing his model from all sides. This model would be animated and then put into the game for the player to control.



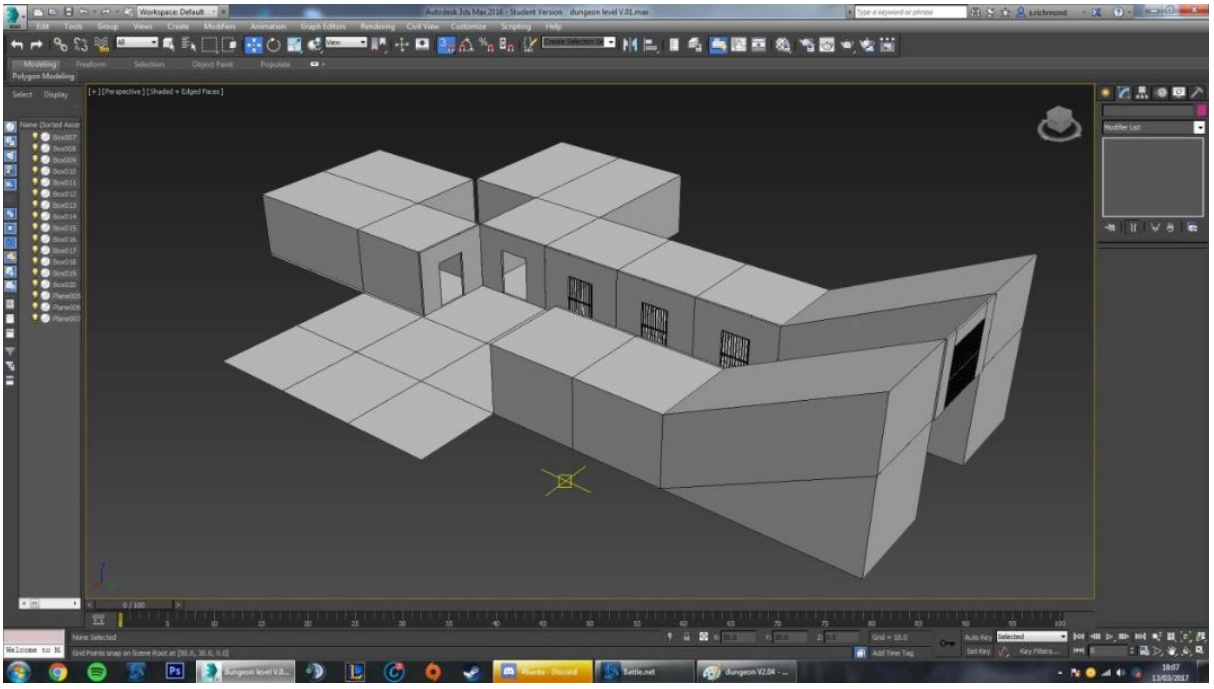
This image also shows the final textured guard model beside his sword and shield that he shall wield in the game.

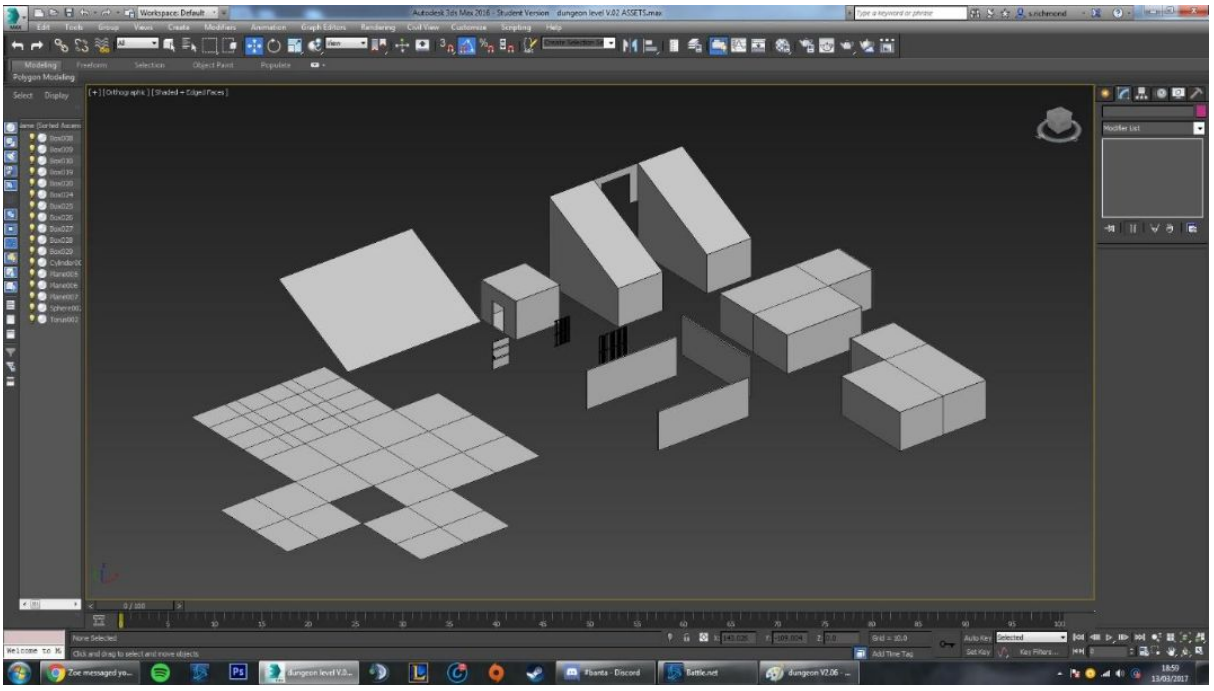


Pictured is also a render of a boss model the player would fight against in the second level of our game. We created this by altering the guards model making him taller and bulkier along with giving the model darker armour. The skull was a reused asset that was model by our assets creator as the team felt like the asset could work well on a model.

The following image also shows a high-quality render of the textured old man model from the front and back view.

Environment models



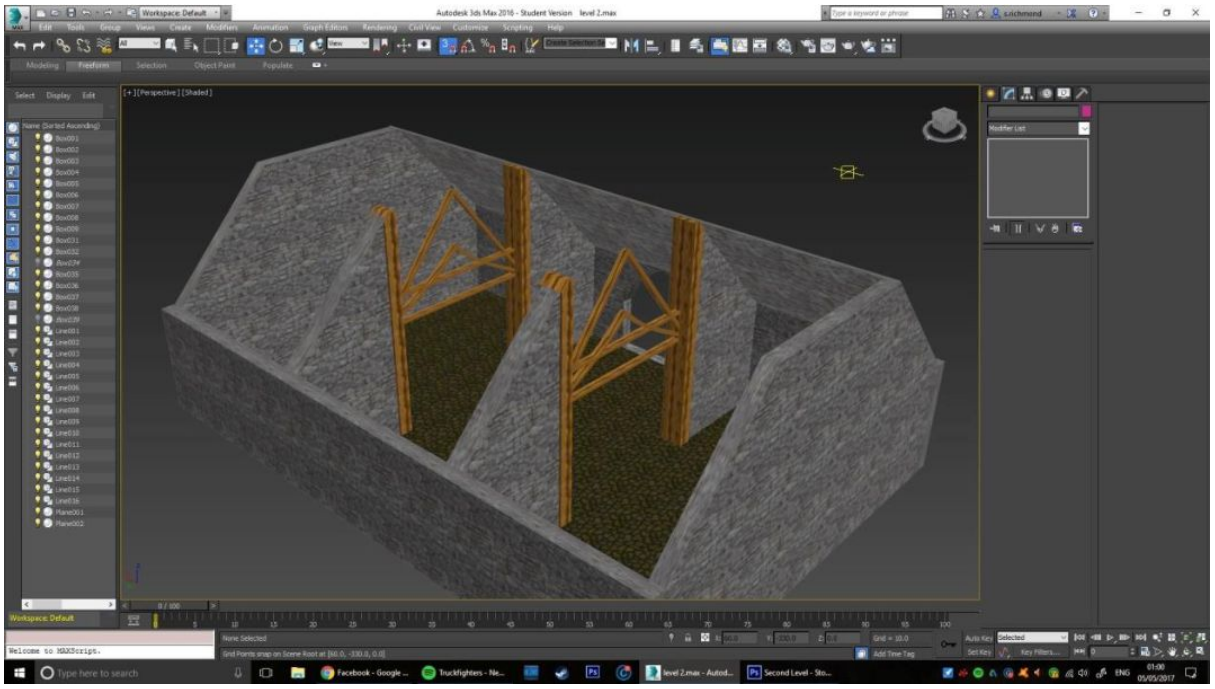


The images above show the initial stages/start of the first section which will be the ‘Dungeon’ scene. This is where the player will start. The entirety of the first section was made up in pieces, so each part of the structure is separate, e.g. cells, cell doors, wooden doors, walls, etc. The reason for this is to make the game somewhat procedural so that we can reuse pieces to make up new sections of the map instantly and efficiently.

This image shows the fully textured dungeon cell's area in the game where the players start with characters and textured assets also implemented.

In the two images to the left are both fully textured store rooms that players can enter through opening the doors. From here players can search boxes and barrels for potions to go into their inventory.

Above is the textured the tunnel players walk through to access all areas in the game.



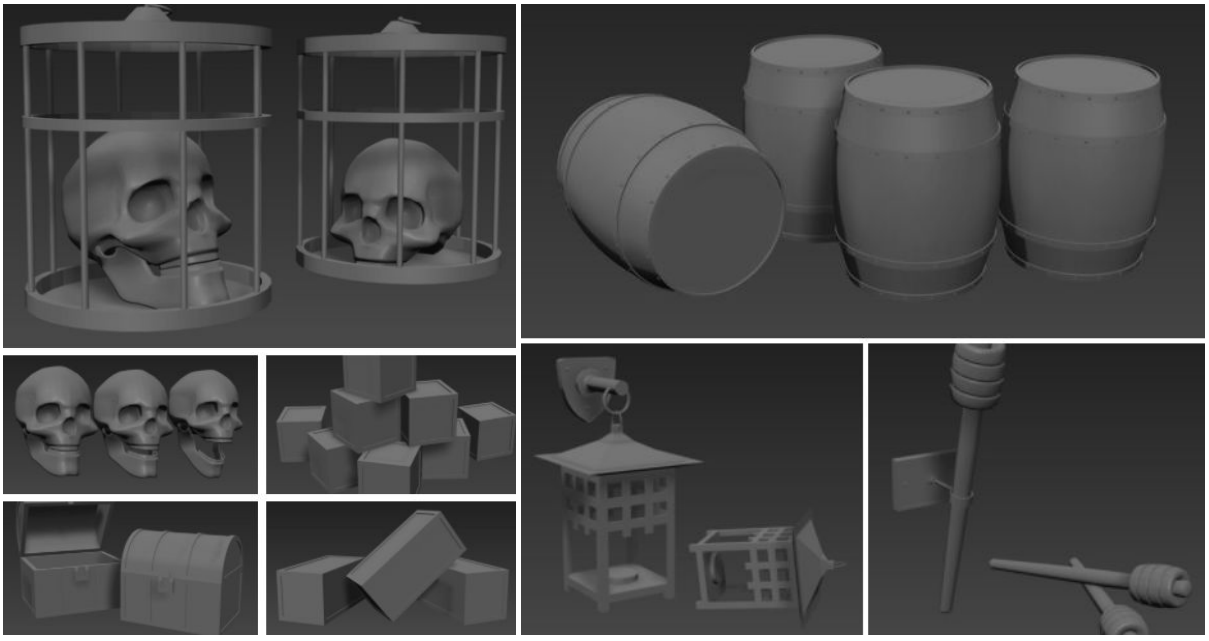
Above is the puzzle room modelled in 3DS Max showing the overall layout of the room and a doorway where players can access the room.

As seen above is this room implemented into the game with the puzzle implemented into the room via the levers and the assets on the wall. Upon completion of this puzzle players can then access the boss room.

The final room players will explore is the boss room where players must defeat the boss and his minions in combat and make use of the potions they obtained previously in the store room. For this room we aimed to make it look like a grand hall with a throne placed at the end and pillars planted around the room to weave in and out of to escape the enemies

Asset models

Assets models were created to make the environment more interesting and alive and to help build the atmosphere of the dungeon. A good set of objects with professional textures is fundamental in order to convert the game's virtual world into an immersive environment. This group of assets includes both intractable objects and just decorative ones. The purpose of this job is to create furniture and decoration as realistic as possible, to match the game style and scenarios. This section will be explained in detail in the art chapter of this document.



As shown in the above image is a skull asset that the programmers would scatter around the map either in groups or alone to give the feeling of dread to the environment and signify the danger of escaping this place. The skull asset was also put into a cage that would rests on shelfs to further exploit the feeling of dread. Moreover, a variation of this skull was used as the head of the final boss.

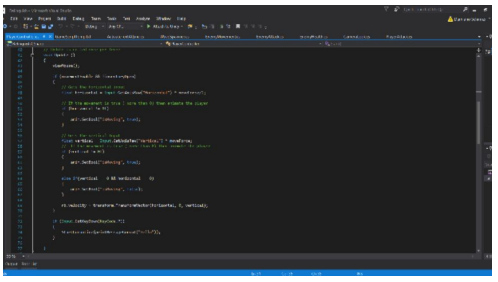
Above are two variations of torches that would be placed on the dungeon walls that would light the environment up and create a dense atmosphere that boded well for our overall game. Along with the skull and the torches we can see a barrel model that would accompany simple box models to populate the corners of the environment. Chests were also created so the player could open them to find potions, keys etc for their inventory.

In addition to these objects there are many more models with its textures. However, as the asset modelling was the main part of my contribution, the whole set of pictures and details can be seen in the chapter 4 of this final report, which is entirely dedicated to this issue. Some of the assets of the mentioned set are gates, tables, stools, a throne, banners and more.

Mechanics

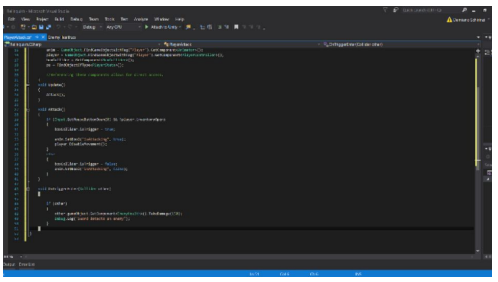
The Player

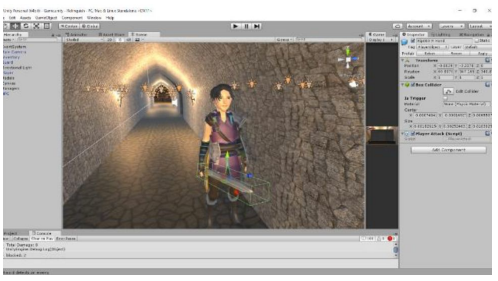
Player Movement



Our game is controlled using WASD and gives the player a 3rd person view of the action. The player can navigate the game world using these simple controls. The player can hold down the shift button to sprint. The player controller script controls the player's ability to move. Here we can tell the code when it should be allowed to set the Booleans of the animator. If the code detects any horizontal or vertical movement then animators "isMoving" Boolean will be set to true which will then begin to play the animation.

Player Combat

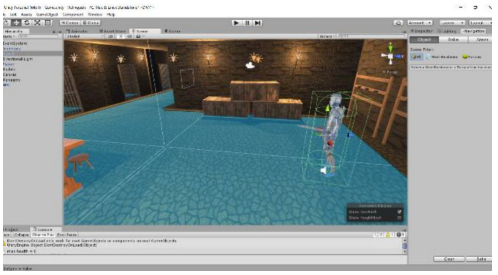


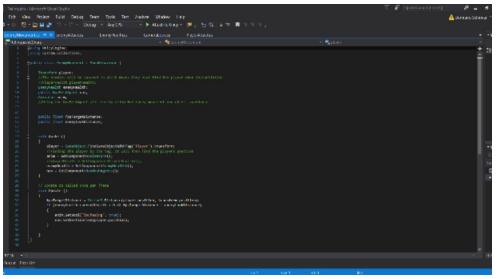


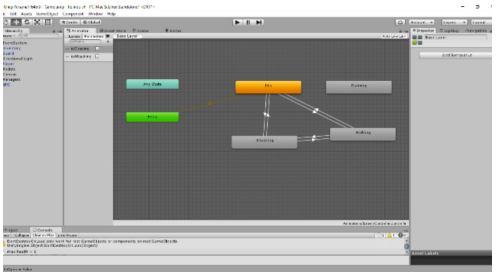
Relinquish features a combat system which is one of the core mechanics of the game. It adds challenge and tension to the project. When creating our combat system, we wanted to provide something that was both easy to master and fun to use. The combat is set to the mouse button, when clicked the character will swing their sword. The character's sword has a Box Collider attached to it. When the player has not clicked anything, this collider will be set as a collider. When the player does click and the character is not already attacking the swords collider will be set to a trigger. This trigger will then enter the enemy's collider and find the enemies game tag. If the tag is correct ("Enemy") the player will deal damage to the enemy. This damage can vary with different enemies. The Attack method is being called in the Update method. This means unity will check if the attack methods if statements requirements have been met. If so, the code will execute and the player will attack.

Enemy AI

Enemy Movement

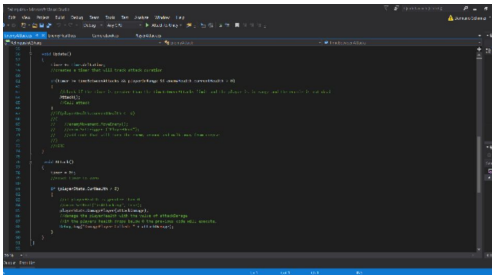






The enemy's movement is controlled by the NavMeshAgent which is restricted to the NavMesh map. Using a NavMesh was a very good choice for us, it allows us to create smart and sophisticated AI which can navigate the map simply that avoid static objects. The enemy's movement script set the player as its target, when the player is within range of the enemy, the enemy will begin to move towards the player character. It will do this in a way which will save it as much time as possible and avoiding all static objects and the same time. The enemy's movement script will check whether the enemy is moving or not, if it is it will access the animator and set the isChasing Boolean to true. This will then start the enemy AI's walking animation and it will zero in on the player. If the enemy is not moving it will be in its default animation, idle.

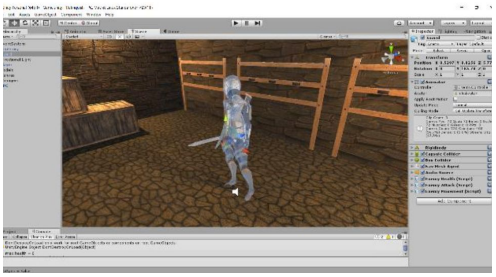
Enemy Attack

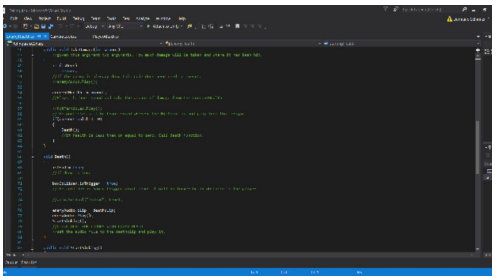


Enemy attack is the second piece of our combat system. Obviously if the player can attack the enemy, the enemy must be given the ability to attack the player. The way our combat works is like the players. The enemy will obviously seek and chase the player if the player is within range. When the enemy has caught up to the player it will stop and begin to attack. The Attack method is called in the update method if the correct criteria is met, the enemy can attack, the enemy is alive and the player is in range. The Attack method code is then executed if the player is alive. It simply damages the player; no box collider is needed because we already know the player is in range of the enemy's attack.

Enemy Health

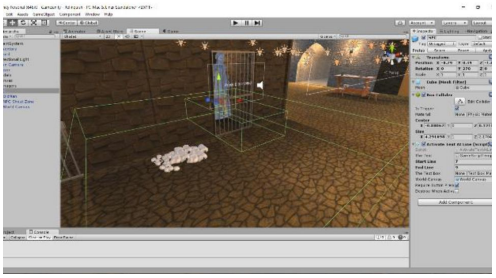
The enemy health scripts control multiple things, obviously, the health of the enemy but also its death as well as when it should start to sink and be destroyed. The enemy's current health is set to the starting health, as this is a public variable we can alter it in the unity inspector allowing us to give different enemies different health values. This in turn will increase the level of difficulty for the player. The take damage method within the enemy health script decrements the amount of current health by the amount of damage taken. This amount damage can be entered as an argument wherever the enemies take Damage method is called. The death method will check if the enemy is dead using a Boolean, it then sets the collider as a trigger so it can sink through the floor and calls the start sinking method. This method is again in the enemy health script and makes the enemy game object sink through the floor and be destroyed.





Dialogue system

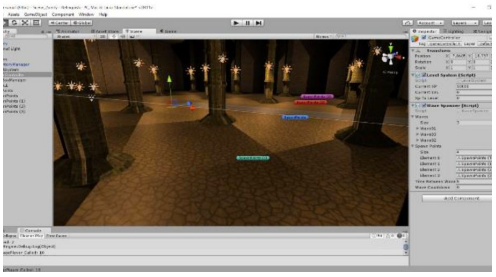
The dialogue system in our game is used to explain the narrative to the player as well as give them quests and quests items. The player can interact with NPC's to begin speaking to them. This system is very easy to set up within the unity inspector and can read multiple lines from one text script. The way I have coded it allows the script to use one text script for multiple characters. This mechanic improves player interaction and makes the game feel more like a classic role playing experience.

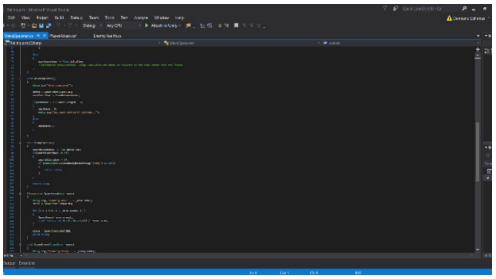


The system features a shout zone and a talk zone, if the shout zone is triggered the NPC will perhaps call the player over or give them an object, this trigger will then be destroyed. The talk zone trigger is interactable; the player can step inside and press E to speak to the NPC. This will then give them information about a quest and allow them to embark on their quest, this trigger will not be destroyed so that the player can once again speak to the NPC if they have forgotten their objective. This script uses the TextBoxManager script which loads the correct text script, makes the current line line 0 and the end line whichever I set it to in the inspector. It will then enable the text box and proceed to print the lines.

Wave Spawning

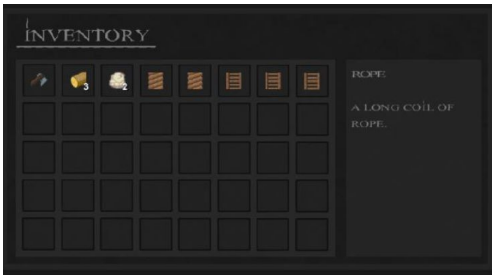
In the final level of the Relinquish demo I have created a wave spawning system. When the player enters the room the system will be triggered and enemies will start spawning at either side of the boss. The boss will then start walking towards the player and begin attack as the guards stand and watch. This makes for a very epic battle and delivers a great ending to our game. This script was quite complex but in turn was easy to use within Unity. The script will instantiate the prefab I set as the current enemy for that wave. I can change the number of waves and give them different enemies and number of enemies per wave. The enemies will spawn randomly from the different spawn points. I am very proud with this mechanic and am going to work on it a lot. The wave completed method will deal with what happens once all the enemies are dead. It checks the current waves number and add one to it. I can set this to loop which allows for endless waves of enemies. The wave system has a countdown, I can display this number using unity's UI. The update method will check if enemies are alive, if they are not wave completed will be called.





Inventory System

In our game you will discover an inventory system. This allowed us to have a lot more interaction in the game due to us introducing items which can be stored in the inventory. The inventory system is highly interactive with the ability to drag/ move items into different slots - Allowing you to rearrange the inventory, and also use/ consume items such as health potion.



The Inventory system is a GUI layer so it will not interfere with the world space.The inventory can be accessed at any time by pressing the button "i" on the keyboard. This is told to you at the beginning of the game, and also on a control option found in the pause menu and title screen. As well as the ability to store and interact with items within the inventory, it also displays a tool-tip feature which gives a brief description of the selected item, and how it is able to be used - Such as double clicking a health potion in order to consume it.

Interactive world space

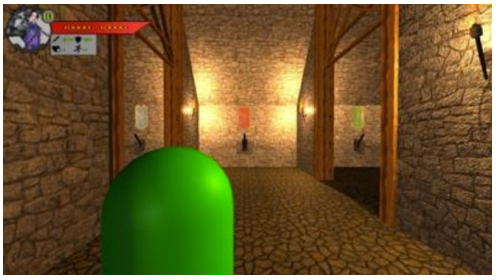
From the start of the game you are instantly introduced to discover an interactive world-space. Due to the player starting in a jail cell, he will be locked inside. It is not until the player approaches the cell door that he will be prompted with a message that states that they need a key.

Upon obtaining this key the player is able to press the "E" button on their keyboard for them to open the gate. This mechanic is used throughout the game which we have incorporated within other system to allow further interaction. Objects such as levers, crates, doors, all share this mechanic - Allowing the player to feel as if they are using the interactive objects themselves.

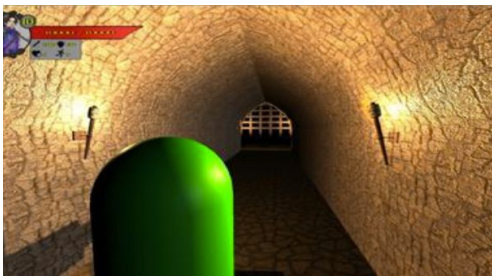
The way the script works is by the use of a GUI layer to prompt an onscreen message, and also a Raycast from the player's position to the direction they are facing. The distance of which that they are able to interact with these objects is easily adjustable. We have chosen the interact distance to be a believable, and suitable distance for a game of this size.

Puzzle Room

The puzzle room is simple; you are given in-game clues which you must solve in order to be able to proceed to the next area. By the use of using the interact script we have included a room which features levers that must be pulled in the correct order. The lever system has a checking system in place that will reset all the levers if the next lever pulled in the sequence is wrong. We decided to include this feature as the player would have to solve the clues and yet again interact with the world space.



Upon all the levers being pulled in the correct order, a gate will open in the connecting tunnel passage which will allow them to continue to the next level.



Searchable objects

A way that we have combined the interact system, and inventory system was a terrific decision. Within the world a player will find crates/ barrels/ chests which will have the option to be searched. This feature involves the interact scripts by being able to press the button "e" on the keyboards which will then allow the player to search these objects. Rather than having a static item in play which the player will automatically gain each time, they will in fact have a 1/3 chance with our random number generator to actually receive anything.

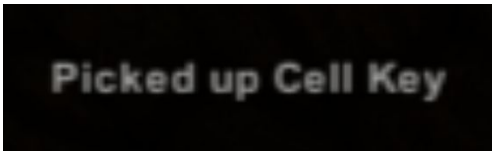
```
// Generates random item to add to the player inventory
public void itemsFound()
{
    // If the object has already been searched then
    if (objectSearched)
    {
        PC.printMessage("You have already searched this chest");
        return;
    }

    // Adds item to player inventory
    else
    {
        objectSearched = true;
        itemToGive = Random.Range(3, 5);
        inv.AddItem(itemToGive);
        itemToGive = Random.Range(3, 5);
        inv.AddItem(itemToGive);
        PC.printMessage("You have found something!");
    }
```

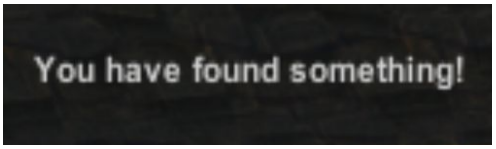
This is where the inventory script comes in. If the player is lucky enough to land on that 1/3 roll, then they will find an item within the chest. Not only is the chance to receive an item randomly generated, but so are the items received too!

Items such as health potions will be automatically added to the inventory upon a successful search making the game-space more fun and interact able as they may not always get what they desire, they have the chance into obtaining what they need.

After the player has searched the chest, they will receive an onscreen message with notifies them whether they have found anything or whether the object was empty. As well as this, the player is not able to search the chest more than once, allowing them to "farm" the items into their inventory over and over again.

On-Screen Messages



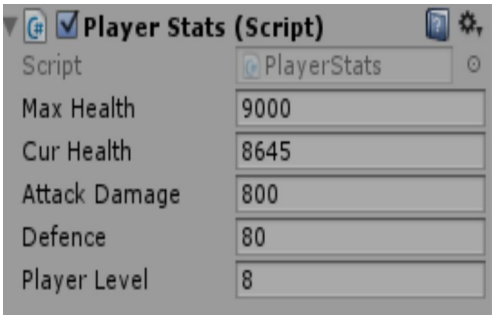


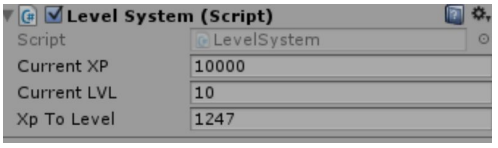
A small feature we have added into the game was the ability to print on-screen messages. This is extremely useful as information the player may not have known before will be printed here. Whether it's searching objects, opening doors, consuming health potions, etc it will all be stated on screen. The script simply prints out what interaction has happened within the game, and what effect it may of had on the player.

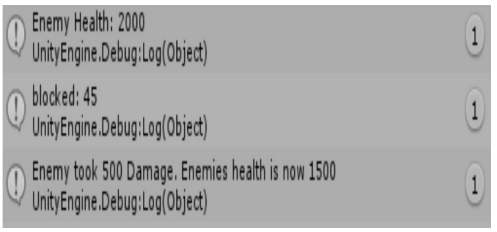
For example, consuming a health potion will print the message saying "Healed for X amount". After the text has been displayed on the screen, it will slowly fade out, allowing the textbox to be reused again.

Level System

A feature involved in many RPG games are level up systems. We decided to add this mechanic into our game as it fit the theme. However, rather than levelling up, the player will level down. Creating an effect which will give the player a more increasing challenge as the game plays on. The story behind the level-down system is within the narrative to the game.







The player will have a Heads Up Display on their screen at all times. The HUD will show the player their current stats such as defence, attack damage, health level, total level, etc.

The level system features an XP system which has a scaling value between levels. This means that it will become harder to level up as there is a higher XP difference between levels. This also scales the attack damage, maximum health, defence level, and so on.

The level system is combined with the combat system allowing the player to only deal his total attack damage at all times. When the player levels down/ up. The attack damage will change so that his output damage is lesser/greater.

A unique mechanic we had put into the level system is a defence level. Depending on the defence level, the user has the ability to block a maximum of 80% of incoming damage. (Preventing it from being too strong). This means that the higher level the player is, the then the better he will be able to defend himself. The math behind this damage reduction is as follows:

(Defence / dmg) * 100) / 2) + defence / 5

Using this system, we have managed to fully balance, and execute a functional, robust, level and combat system.

Story:

The narrative for our overall game was to have the player take on the role of the protagonist named Zephyr who you find to wake in a dungeon cell with a case of amnesia. In the neighbouring cell an old man shall greet you and give the player objectives to help guide them out of the dungeon and later in the game provide the player with side quests.

The player would learn the basic mechanics of the game in the opening dungeon section such as combat, character interactions, puzzles and inventory management. After the player escapes the dungeon with the old man they are greeted with an open world of the kingdom and trek to the kingdoms village to go on quests in order to regain the protagonist's memory.

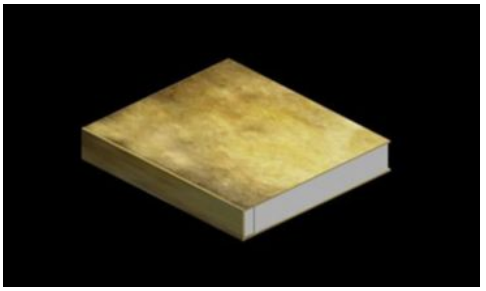
The old man would be revealed to be a sorcerer who is manipulating Zephyr into thinking he is the rightful heir to the throne and must reclaim it. This ultimately makes our protagonist and player believe this and go and tackle the castle until they reach the true king only for the truth to be revealed and have the player be deceived into playing the villain all along.

How we will deliver this narrative is through character interactions as demonstrated in the image below.

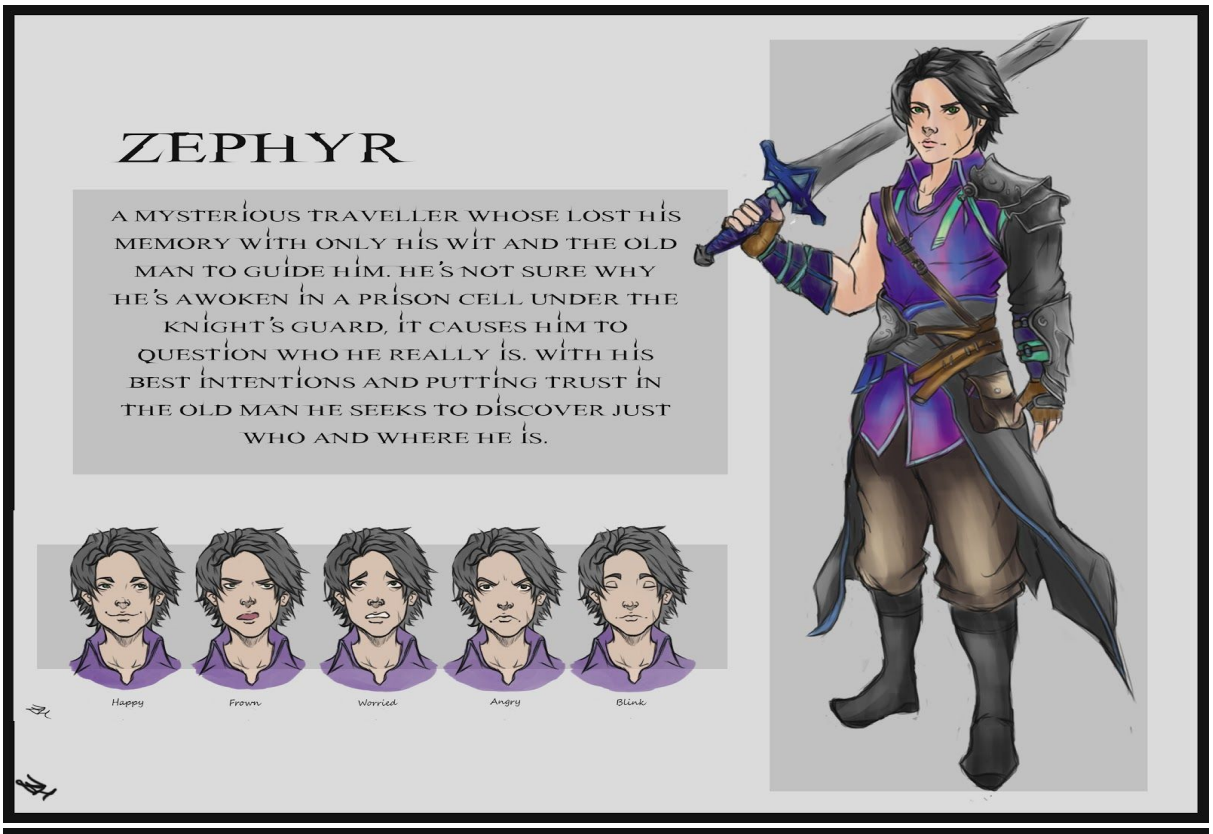
The player will be able to interact with characters to be given context to the story, introductions to NPC's in the world and given objectives to progress through the story.

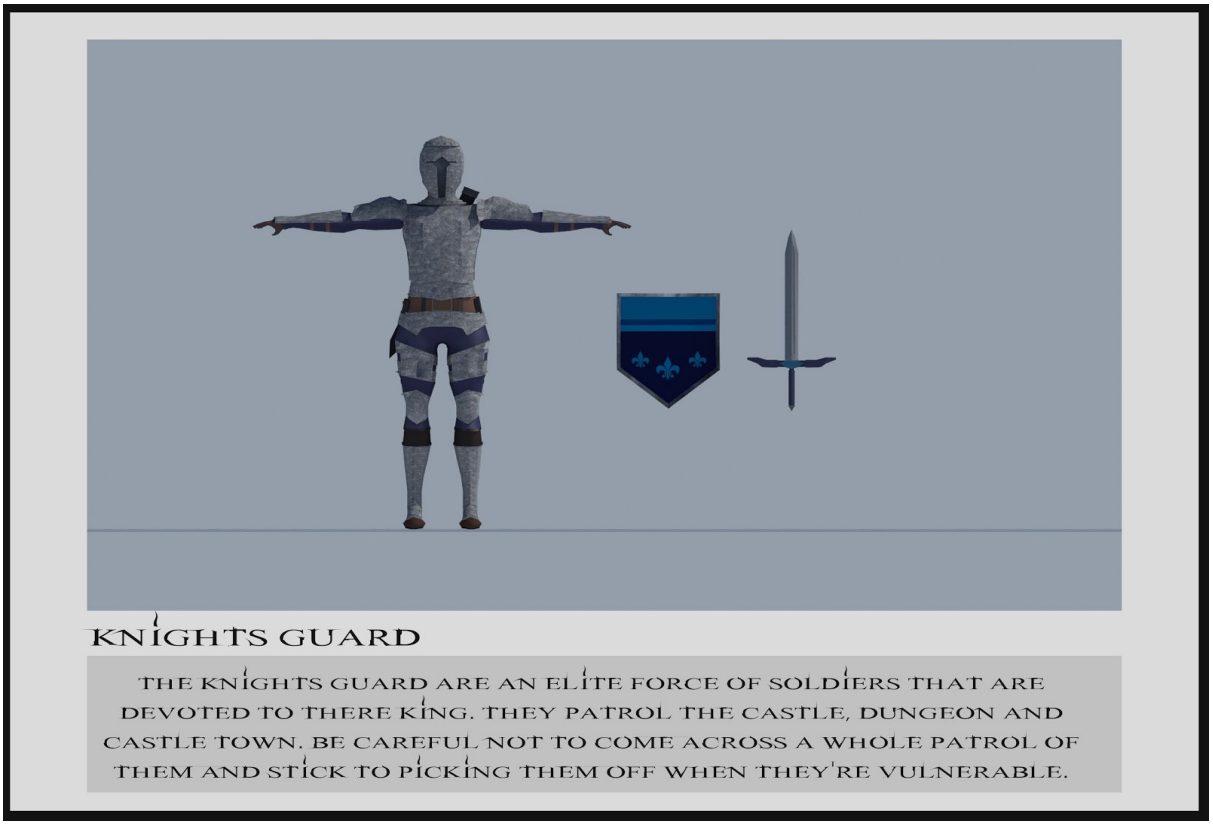
The other way we aim to tell the narrative is through the lore book we've put into the game, as pictured left.

The image is a render of the books model that players can interact with in the game.



The lore book will contain concept art/renders and lore about the characters and world of Relinquish along with dropping hints on how to tackle certain enemies and areas. This also gives players an insight into what they might encounter later into the game giving a better overall picture of the game/world we're trying to build.Below are a series of images that are used in the lore book.





OLD MAN

NAME: SYLUS
AGE: UNKNOWN
A MYSTERIOUS CAPTIVE OF THE KNIGHT'S GUARDS. LITTLE IS KNOWN ABOUT HIS TRUE ORIGINS AS HE IS SHROUDED IN MYSTERY. HE IS ONE OF THE MOST MISTREATED PRISONERS OF THE KNIGHT'S GUARD, IS THIS DUE TO HIS AGE OR A CRIMINAL ACT HE COMMITTED? THIS MAN WIELDS A MYSTERIOUS AURA ABOUT HIM ALONG WITH KNOWLEDGE THAT WILL GO A LONG WAY TO HELP YOU.

GOBLIN

CREATURES THAT ARE BOUND TO DARKNESS. YOU'LL USUALLY FIND THEM HIDING IN DUNGEONS AND CAVES, SOME MAY EVEN WONDER THE LAND AT NIGHT TIME. CRAFTY AND DECEIVING CREATURES, DON'T TRUST THEM AND DON'T LET THEM GANG UP ON YOU!

CASTLE BRIDGE

THE BRIDGE BETWEEN THE CASTLE AND VILLAGE IS OFTEN ONLY TRAVERSED BY THE KNIGHTS GUARD OR VILLAGERS WHEN THE KING DEMANDS AN AUDIENCE.

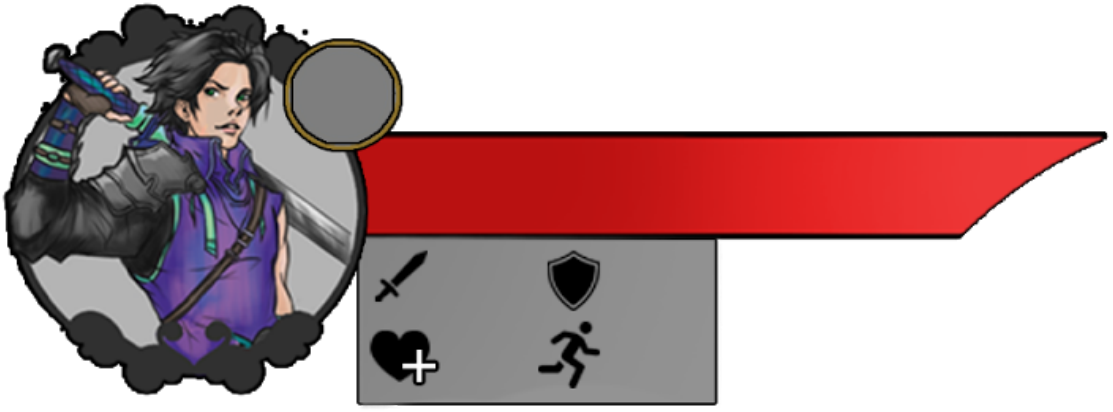
THE BRIDGE HAS SUSTAINED SIGNIFICANT WEAR AND TEAR OVER THE YEARS, BE CAREFUL WHEN TRAVELLING ACROSS IT WHEN THE TIME ARISES.

OLD CAVERNS

THE OLD CAVERNS SPREAD OUT AMOUNGST THE CORNERS OF THE KINGDOM ARE HOME TO TREASURES BUT ALSO TO GOBLINS WHOM SEAK TO PROTECT SAID TREASURES.

HAVING SOME REMEDIES ON HAND WOULD HELP GREATLY IN RETRIVING GOLD.

Interface:

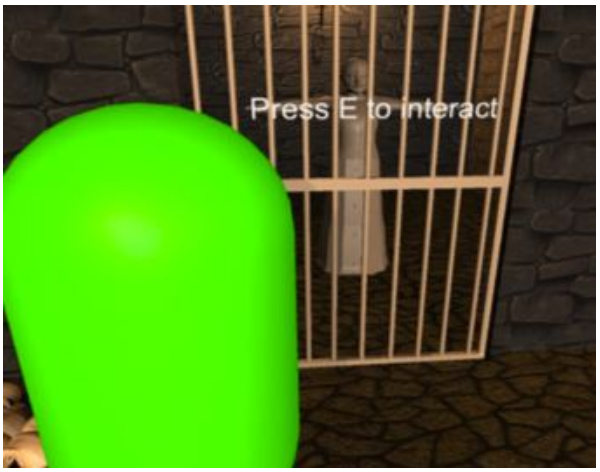


Above is the UI players will see in the top left corner of the screen showing their health, level, character portrait, attack damage and defence level.

In the left we have the character portrait. This was used as a good way for the player to identify the protagonist and bring some of the art style from the concept art into the game. We also wanted to include this as the artist did the same thing last semester for some of our team members previous game and it boded well for the aesthetics of the game

In the above image is the inventory menu which can be accessed in game by pressing the key “I”. From here the player can view, manage and use items collected from chests in the game. On the right hand side of this menu is a brief description of the item selected to players are aware of how the item functions. Apart from the inventory, there are three examples of item sprites created for inventory and are also examples of the kinds of items players can find in the game. Key’s unlock doors, green potions restore greater health and red potions restore health.

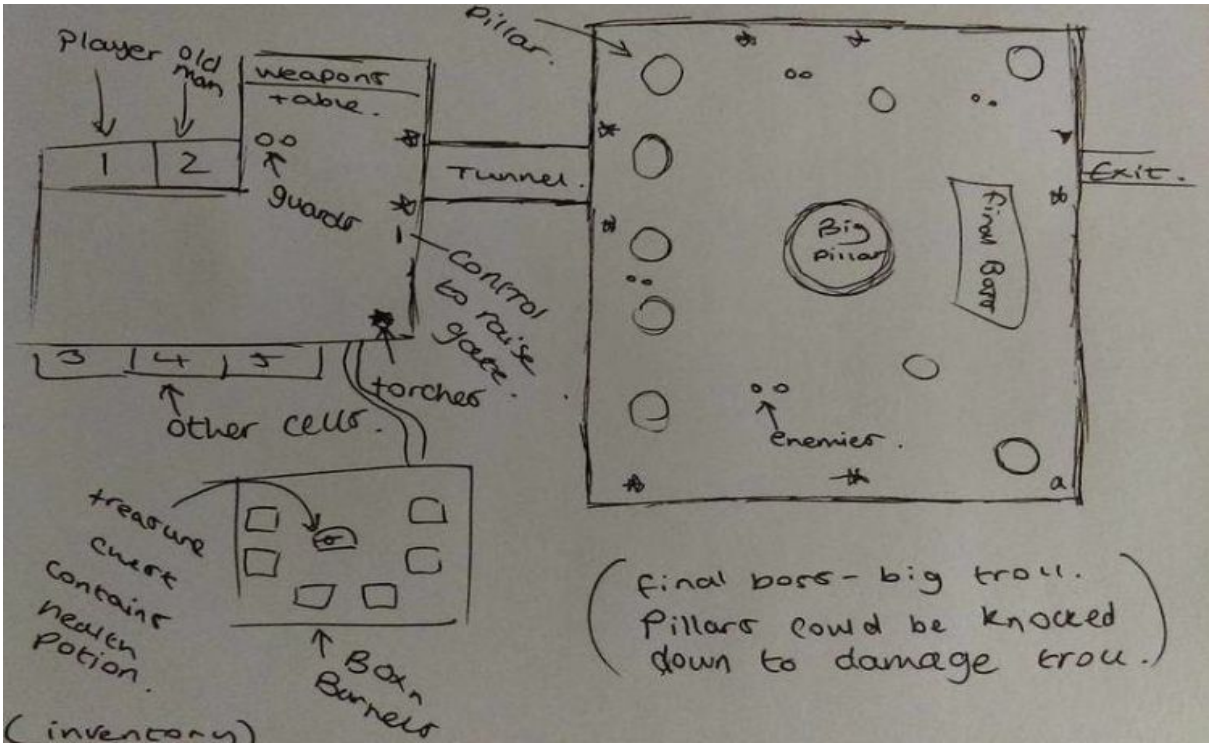
Above is the start menu for our game with the title, “play game”, “Load game”, “Options” and “Quit game” options available. The artist for the game produced a piece of artwork for this screen of the protagonist amongst some flames to give a more dramatic effect. In unity a fire particle effect would also be added which further enhanced this screen. In the below image is the controls screen which can be accessed via the start menu.



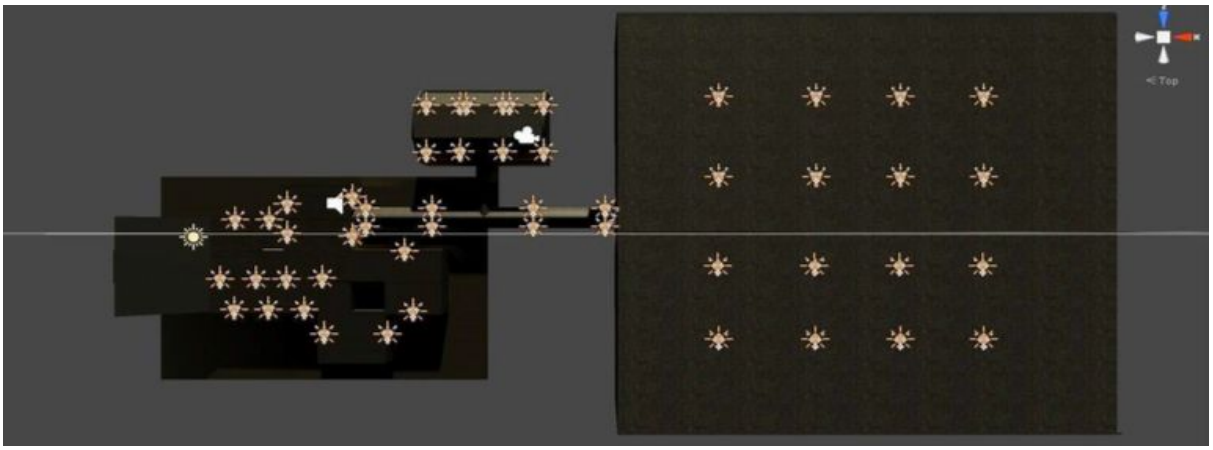
Pictured above are two examples of the UI in game is used. This is either presented as button prompts to interact with a character or to interact with an object to proceed or solve puzzles in the game.

Above is an early example of our game using character interaction. The player engages in conversation with the old man who will introduce the player to the game and give them his objective.

Sandbox plan:



Above is an image of a drawn map plan of the dungeon's cells room (top left), storeroom (bottom left) where players would open chests to find potions and other items and finally (right) a boss room where players would encounter a giant boss to finish off our game demo and offer a combat challenge, this would also put the items the player had found earlier in the store room to use.



In the following image is the finished and final map plan of our game with the store room connected to the dungeon cell room and a puzzle room added which is the top small room above the cell room. You can also see where lighting has been placed in the room and that our layout flows nicely between rooms thanks to the tunnel modelled.

Enemies are in the cell room and boss room (biggest room far right). The store room is the location of item pickups and the player starts at the far end of the left side of the map in a cell.

Technologies:

Unity 5.6.1f[14]

When planning Relinquish the team was left with the choice of which Editor to use. We were deciding between Unreal and Unity. In the end, we decided to create our game using Unity, this was mostly decision of the programmer team who are very confident using Unity editor and its main scripting language, which is C#. With the latest update patch of Unity (Unity 5.6.1f), the software has been provided with multiple improvements to folder structure and organisation. This was a large factor when considering our choice as it would help the programmers work together. One huge drawback to using the newest version of Unity was the fact that computers at university did not have the latest version. This meant that when we were trying to work on our projects at university we were not able to due to the version being downgraded and scripting API becoming missing.

Microsoft Visual Studio[15]

We used Microsoft Visual studios as our chose IDE, on this we scripted in C#. Again, we did this because the programmers are both very comfortable with this IDE. Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging. Things like code completion and debugging are handy tools which save us time by completed syntax and rooting out errors with ease.

TeamViewer[16]

TeamViewer is a very popular and handy program which allows remote control of another person's computer. As the programmers could not meet up daily we instead connected on TeamViewer and helped each other code. This was very useful as it allowed us to pick up on errors and mistakes quicker as well as solve problems and create new mechanics. We thought it was best to both install this program in case we ever needed a joint effort on a certain part of the game. In the end, we believe using TeamViewer helped us.

Slack[17]

Slack was our main Archiving platform and it also helped us to communicate, we used slack to upload files as well as keep backups of everything we have done. In terms of working on and developing the project, the programmers would constantly be uploading the latest version of Relinquish. We would organise this by incrementing the version number every time we uploaded a new one. For example, Relinquish 1.22, Relinquish 1.23, etc. This was an extremely helpful tool which saved us a lot of time.

Photoshop[18]

Photoshop was used in our project to create textures and concept art. The program is very popular and for good reason. We chose to use it because our artists deemed it best and are all very confident with using it. Photoshop is an effective but simple program which allowed our artists to work efficiently and produce marvellous work.

3DS MAX[19]

We chose to use 3DS MAX as our computer graphics program as it is a very robust modelling toolset with a huge library of different modifiers which can made the modelling process easier. Again our artists and modellers are comfortable using this software and we wanted to make the making of our project suitable for everyone's level of skill.

Adobe Audition[20]

This Audio mixing software was used by our sound designers to create the amazing sounds in the game. This program is great as it allows the sound clips to be exported as .wavs, which can be directly imported into Unity.

Chapter 4:
Development

One of the project requirements of the individual projects was the development of a programming part. As it is explained before in this report, for my programming task I chose to do a camera controller and a artificial intelligence for a final boss character which included the movement, the attack and the health scripts.

Camera controller

The following class shows the script created to control the main camera of the game, as it is a third person based video game, the camera follows the main character and rotates around him. This scripts also solves the problems that may occur because of the movement of the camera, such as the camera crossing through the walls or turning face down because of the vertical rotation.

```
public class CameraController : MonoBehaviour
{

        public Transform target;
        public float distanceFromTarget = 15f;
        public float clippingDistance = 15f;
        public float clippingAdjustPos = 0.5f;

        public float vertClampMin = -30.0f;
        public float vertClampMax = 90.0f;

        public Vector3 cameraPosOffset = Vector3.zero;

        private float horizontalRot = 0.0f;
        private float verticalRot = 0.0f;

        public bool movementEnabled = true;

        void LateUpdate()
        {
                AdjustToEnvironment();

                if (movementEnabled == true)
                {

                        horizontalRot += Input.GetAxis("Mouse X") * 2f;

                        verticalRot -= Input.GetAxis("Mouse Y") * 2f;

                }

                verticalRot = Mathf.Clamp(verticalRot, vertClampMin, vertClampMax);

                Quaternion rot = Quaternion.Euler(verticalRot, horizontalRot, 0);
                transform.rotation = rot;

                Vector3 pos = target.position - (rot * Vector3.forward * distanceFromTarget);
                pos += cameraPosOffset;
                transform.position = pos;

                Vector3 targetEuler = target.eulerAngles;
                target.rotation = Quaternion.Euler(new Vector3(targetEuler.x,
                                                transform.eulerAngles.y,
                                                targetEuler.z));

        }
```

```
        void AdjustToEnvironment()
        {

                Vector3 camForward = transform.forward;
                Vector3 InvCemForward = camForward * -1;

                Vector3 origin = transform.position + (camForward * distanceFromTarget);

                float dist = clippingDistance;

                RaycastHit hit;
                if(Physics.Raycast(origin, InvCemForward, out hit))
                {

                        if (!hit.transform.CompareTag("Team 1") &&
                            !hit.transform.CompareTag("Team 2"))
                        {

                                if (hit.distance < clippingDistance)
                                {

                                        dist = hit.distance - clippingAdjustPos;

                                }
                        }
                        else
                        {

                                dist = clippingDistance;

                        }
                }
                else
                {

                        dist = clippingDistance;

                }

                distanceFromTarget = Mathf.Lerp(distanceFromTarget, dist, Time.deltaTime * 10f);
        }
}
```

In the previous script, after declaring the needed variables we found the *LateUpdate()* function, with the code to move and rotate the camera and also limitate its rotation:

The *horizontalRot* and *verticalRot* variables which store the movement of the mouse. The function *Mathf.Clamp()* clamps the camera in the Y axis between a maximum and a minimum limits so the camera can't make a full vertical rotation.

The quaternion is a way to represent rotation in 3D space. This rotation is made according the mouse movement using *Quaternion.Euler(verticalRot, horizontalRot, 0).*

*Vector3 pos = target.position - (rot * Vector3.forward * distanceFromTarget);* makes the camera rotate around the character from a determinate distance (*distanceFromTarget*).

The last two lines make the character rotate in the Y axis according to the movement of the mouse.

On the other hand, the *AdjustToEnvironment()* function contains the needed code to adjust the position of the camera depending on whether there is an obstacle between the character and the camera or not.

The first to lines of the function basically create a vector in the direction of the character to the camera and the next line makes the character the origin another vector. The variable *dist* refers to the base distance from the camera to the character.

Next, the function detects the objects around the character by raycasting from it. The raycast calculates the distance of the object from the character (*hit.distance*) and if it is smaller than the base distance (*dist*) equals the last one to the hit one minus an adjustment (*clippingAdjustPos*) so the camera is not moved to the exact position of the object.

The *Mathf.Lerp()* function makes the camera movement towards the character look smooth so it is still possible to see through the objects if the mouse is moved too fast. The camera movement speed can be adjusted modifying the amount multiplied by *deltaTime*.

Final boss AI

- ### Movement

The following script controls the movement of the final boss and the player finding.

```
public class BossMovement : MonoBehaviour {

    Transform player;

    BossHealth bossHealth;
    NavMeshAgent nav;

    public float fpsTargetDistance;
    public float bossLookDistance;

    void Awake ()
    {
        player = GameObject.FindGameObjectWithTag("Player").transform;

        bossHealth = GetComponent<BossHealth>();
        nav = GetComponent<NavMeshAgent>();
        }

        void Update ()
    {
        fpsTargetDistance = Vector3.Distance(player.position, transform.position);
        if (bossHealth.currentHealth > 0 && fpsTargetDistance < bossLookDistance)
        {
            nav.SetDestination(player.position);
        }

    }
}
```

The final boss character is spawned by the time the player enters the final level which means it must find the player when instantiated.

Using the *NavMeshAgent* will easily allow for final boss character's movement and object avoidance. The final boss is able to find the player using the following line: *player = GameObject.FindGameObjectWithTag("Player").transform;* which allows the final boss to find the player by his tag.

The *Update()* is called once per frame, setting the final boss destination point using *nav.SetDestination(player.position).*

- **Attack**

  The following script controls the attack system of the final boss:

```
public class BossAttack : MonoBehaviour {

    public float timeBetweenAttacks = 0.5f;
    public int attackDamage = 10;

    GameObject player;
    PlayerStats playerStats;
    BossMovement bossMovement;
    BossHealth bossHealth;
    bool playerInRange;
    float timer;

    void Awake ()
    {
        player = GameObject.FindGameObjectWithTag("Player");
        playerStats = FindObjectOfType<PlayerStats>();
        bossHealth = GetComponent<BossHealth>();
    }

    void OnTriggerEnter(Collider other)
    {
        if(other.gameObject == player)
        {
            playerInRange = true;
        }
    }

    void OnTriggerExit(Collider other)
    {
        if (other.gameObject == player)
        {
            playerInRange = false;
        }
    }

    void Update()
    {
        timer += Time.deltaTime;

        if(timer >= timeBetweenAttacks && playerInRange && bossHealth.currentHealth > 0)
        {
            Attack();
        }
    }

         void Attack()
    {
        timer = 0f;

        if (playerStats.CurHealth > 0)
        {
            playerStats.DamagePlayer(attackDamage);
        }
    }
}
```

In the Awake() function, *player = GameObject.FindGameObjectWithTag("Player");* finds the player object using his tag. *playerStats = FindObjectOfType<PlayerStats>();* pulls the player health script off of the player, stores and adds a reference to it, this will improve performance as opposed to constantly searching for the script.

In the function *OnTriggerEnter(Collider other)* If something that is not the boss gameObject enters the collider it checks if whatever collided is the player, if so, *playerInRange* is equal to *true.* The next function, *OnTriggerExit(Collider other),* does the opposite, checks is whatever left the collider is the player, to set *playerInRange* to *false.*

In the *Update()* function, *timer += Time.deltaTime;* creates a timer that will track attack duration, then the condition check if the timer is greater than the *timeBetweenAttacks* limit, the player is in range and the final boss is not dead, if so, calls the *Attack()* function.

The *Attack()* function first resets the timer to 0. Afterwards, if *playerHealth* is greater than 0, that means that is still alive, damages the *playerHealth* with the value of *attackDamage*.

- **Health**

  The following script controls the health levels of the final boss:

```
public class BossHealth : MonoBehaviour {

    public int startingHealth = 100;
    public int currentHealth;
    public float sinkSpeed = 2.5f;
    public int scoreValue = 10;
    public AudioClip deathclip;

    Animator anim;
    AudioSource bossAudio;
    ParticleSystem hitParticles;
    BoxCollider boxCollider;
    bool isDead;
    bool isSinking;

    void Awake ()
    {
        bossAudio = GetComponent<AudioSource>();
        hitParticles = GetComponentInChildren<ParticleSystem>();
        boxCollider = GetComponent<BoxCollider>();
        currentHealth = startingHealth;
    }

    void Update ()
    {
        if (isSinking)
        {
            transform.Translate(-Vector3.up * sinkSpeed * Time.deltaTime);
        }
    }

    public void TakeDamage(int amount)
    {
        if (isDead) return;

        bossAudio.Play();
        currentHealth -= amount;
        hitParticles.Play();

        if(currentHealth <= 0)
        {
            Death();
        }
    }

    void Death()
    {
        isDead = true;
        boxCollider.isTrigger = true;
        bossAudio.clip = deathclip;
        bossAudio.Play();
        StartSinking();
    }

    public void StartSinking()
    {
        GetComponent<NavMeshAgent>().enabled = false;
        GetComponent<Rigidbody>().isKinematic = true;
        isSinking = true;
        Destroy(gameObject, 2f);
    }
}
```

In the *Awake()* function, GetComponentInChildren will search for all children of the game object until it finds the right type and will then store it as *hitParticles*.

The *Update()* function, is called once per frame and if the object is sinking it will translate the transform down by *transform.Translate(-Vector3.up * sinkSpeed * Time.deltaTime);*

In the function *TakeDamage() if (isDead) return;* is used because if the boss is already dead this code does need need to execute. If not, the function Plays the hurt sound and take the amount of damage from the *currentHealth*. The particles will be transformed wherever the HitPoint is and play from that origin. If *health* to zero or less, the *Death()* function is called.

The *Death()* function makes the collider a trigger which means it will no longer be an obstacle to the player. It also plays the death sound clip and calls the *StartSinking()* function

Finally, *StartSinking()* disables the Navigation mesh agent, sets the *isSinking* variable to true, so the boss corpse starts sinking in the ground and finally destroys the game object after two seconds.

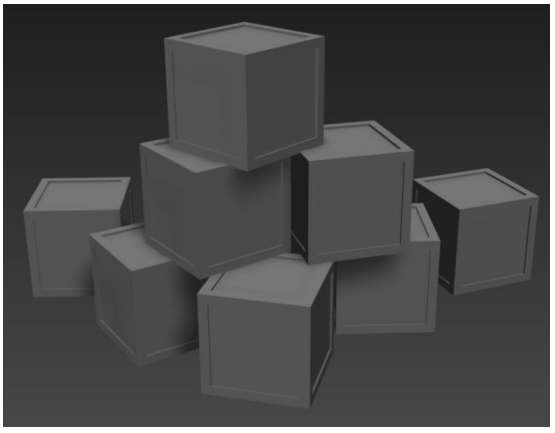
Chapter 5:
Art

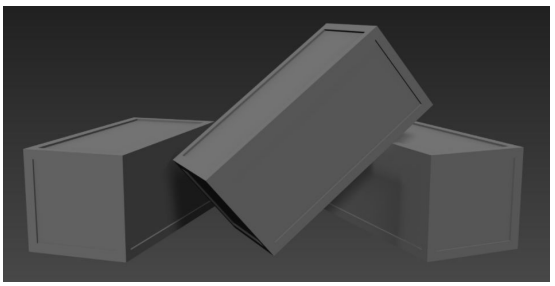
Environment assets

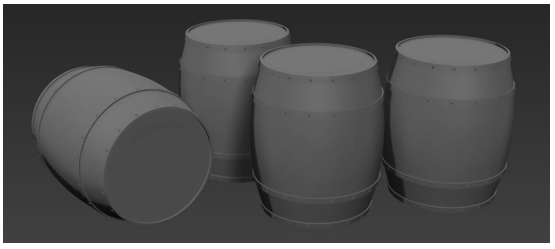
My main job as a 3D modeller and texture designer was creating a set of models to include into the environment to create an immersive experience for the player. The objects had to fit the realistic style of the game and also the medieval genre. During the whole project I have tried to produce every model with a high level of detail prioritizing the visual quality before everything else.

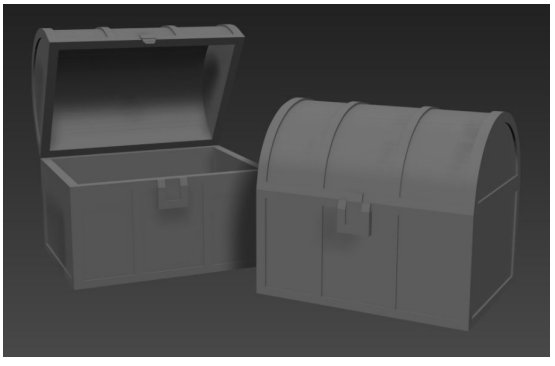
This mentioned set of assets included some wooden boxes and barrels. In almost every medieval videogame, wooden boxes can be found through the map along with barrels.

Apart from that, one of the most common objects in this kind of games are the chests because of the possibility of interaction as a searchable object.In the pictures below some different sizes of boxes, barrels and chests can be found.







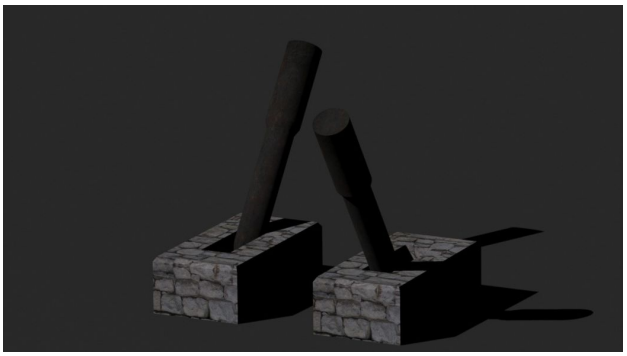


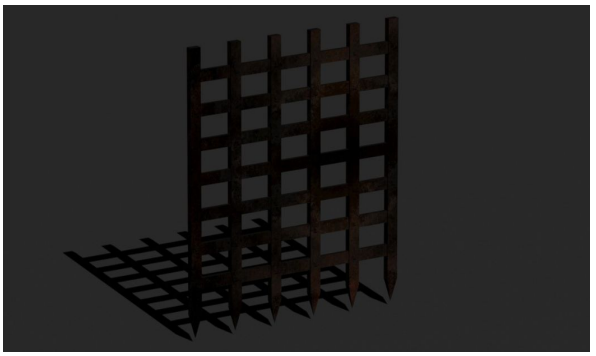
A very important part of the game is the lighting system and the way the caves were illuminated in Middle Age was by using lanterns and torches. Both of this objects were modelled with a support to attach them to the walls. This support was created separately, so the torches and lanterns can be placed on the floor as if they had fallen down.

After including the models in unity, a particle system was added to make it look like a flame inside of them producing the lights.

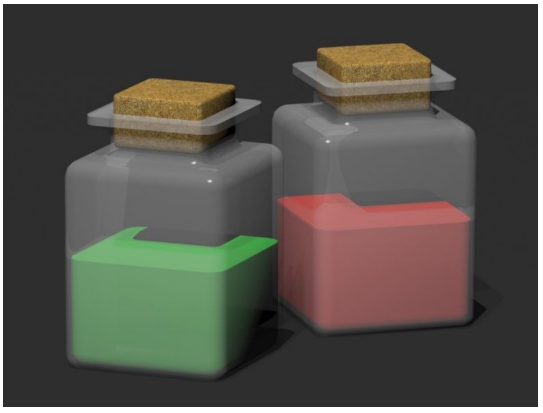
A wooden weapon rack was also required to hold the swords the character and the guards use to fight against each others.

In addition to the mentioned interactive objects, some other assets were needed in order to create puzzles such as levers, gate doors and banners to act as hints. In the images below we can see five different textures created for the banners in the puzzle room. These symbols would ultimately aid in helping the player solve the puzzle and be able to proceed to the boss room. The five symbols stand for “Need”, “Wealth”, “Fire”, “Human” and “Serpent” respectively.





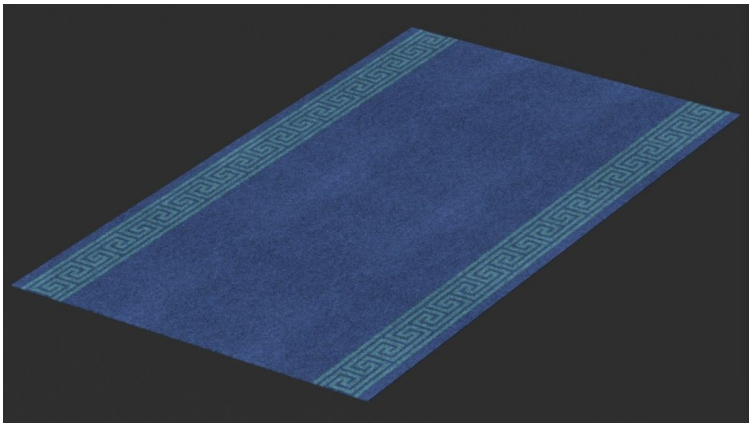
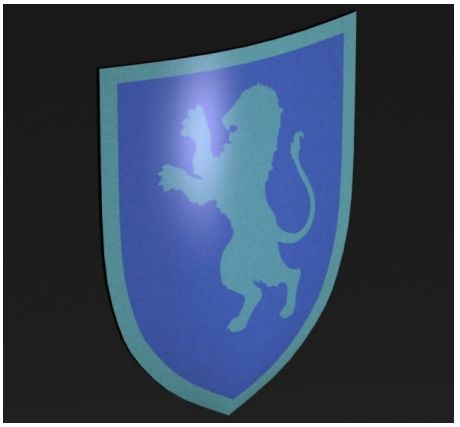
The last interactive objects of the game are the potions. I created two kinds of them, a red one which restores half of the life and a full life restore green one. These are the only two assets that use custom materials instead of textures (except the cork), because it is very hard to recreate transparent glass using 2D image based textures. I tried to use custom materials as little as possible because it can produce errors while importing into Unity.



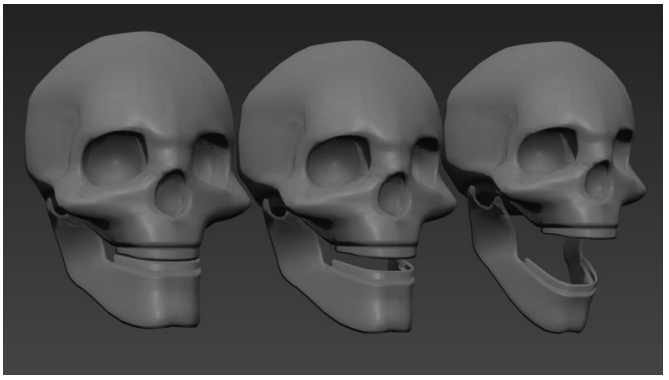
The rest of the objects were just created to make the scene look less empty and more real, including common castle things like tables, stools or a throne, the three of them have been textured using a wood texture, a light one for the first two and a dark one for the throne, this last one also uses a pretty obvious bump map to act as decorative details.

Many banners also populated the second and third level's environment upon the walls using the colour scheme that is associated with the guards you battle in the game.

Moreover, I also modeled some more decorative assets that can be seen in the pictures below. This set includes a carpet and a shield that match the color of the guards armour. I also created some fountains and even food dishes and cutlery, following always the realistic style, as in the rest of the game.

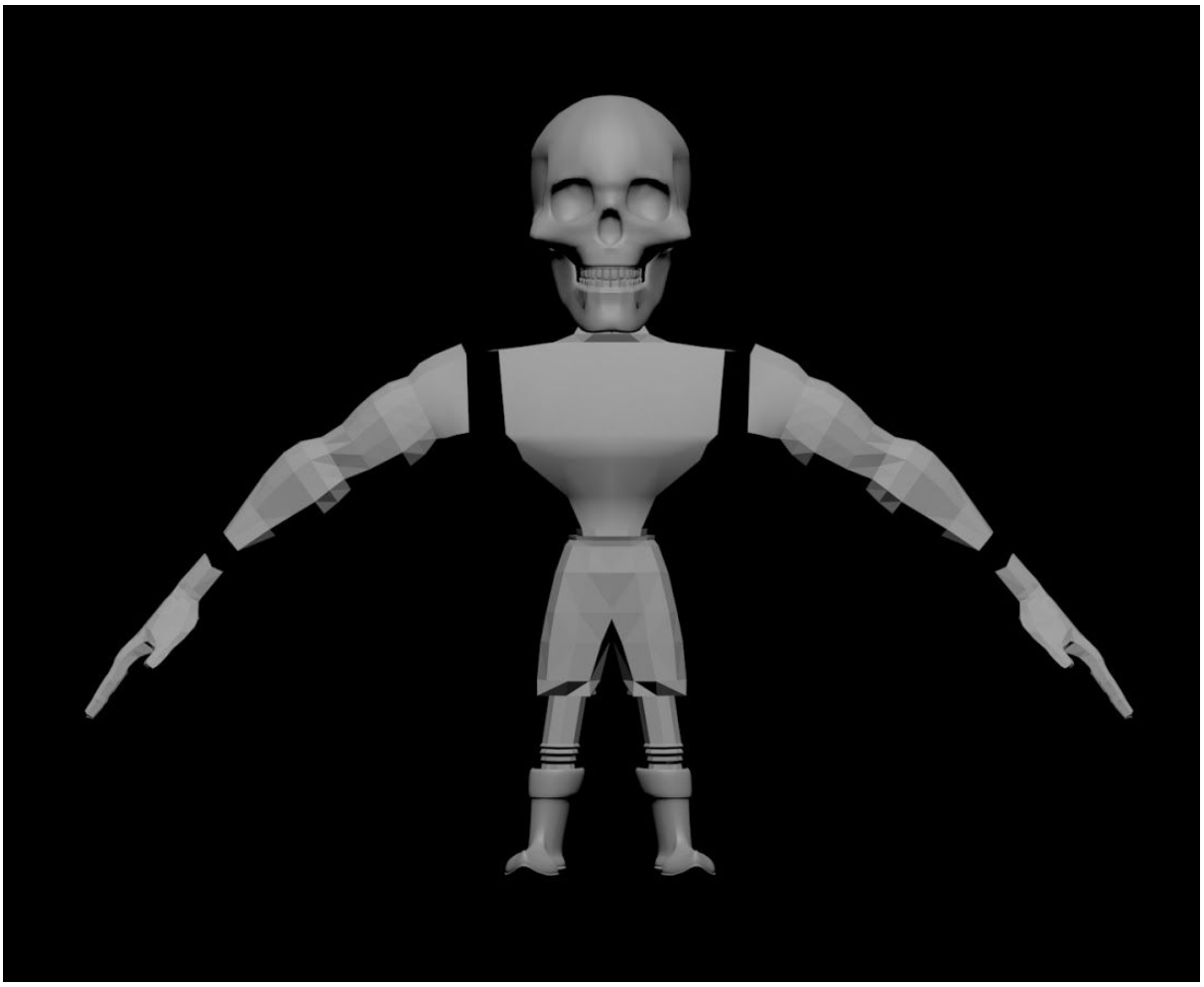
Besides all the previous things, as I made a skull for the final boss head (I speak about it in depth in the next section) I decided to use the same skull as a decorative item, putting some into clusters and some others in caged all over the catacombs.

Final boss character

As I have mentioned before, in the related work chapter, my main source of inspiration in order to design the final boss character was the main character of the *MediEvil* series. From the very beginning our goal was to have a final boss that seemed intimidating and this is the reason for the skull head. He is also way higher than the rest of characters in the game.

The modeling process was divided in modules. Because I wanted to represent the character as realistic as possible I decided to model the different parts of the character separately and put everything together at the end so some parts of the body do not bother me while modeling others. The image below shows this modular modelling.

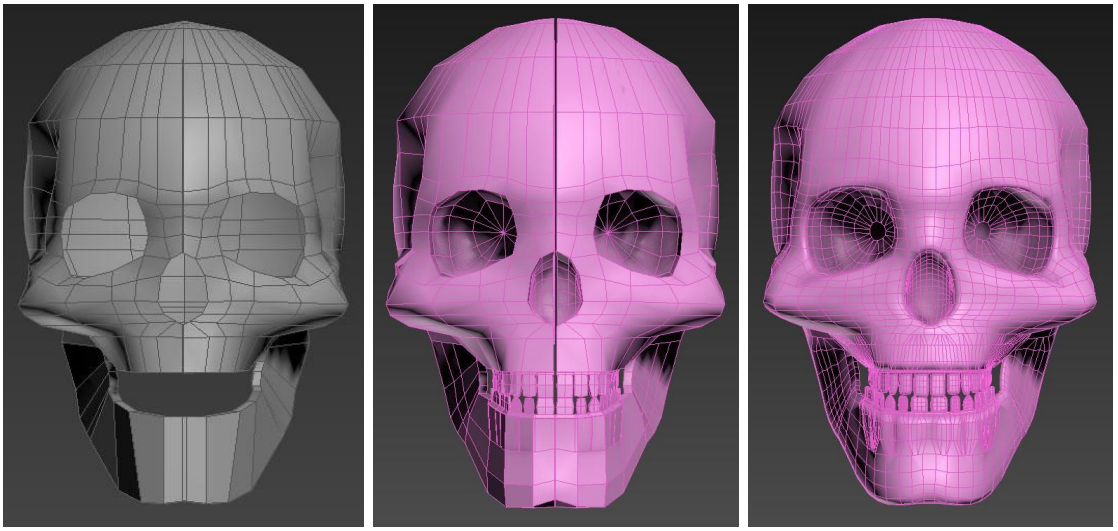


As seen, the different modules were the skull, the hands, the arms and the torso with the legs. Every mentioned part was modeled using symmetry, that means that only an arm and a hand were created and also only half of the skull and the body, the rest were mirrored.

As I mentioned before, the final version of the character was changed a bit because the team found that putting a big armour on the character would make much more sense. The following pages show the full process for the making of the character, including the modular modelling, the first version and the re-edit.

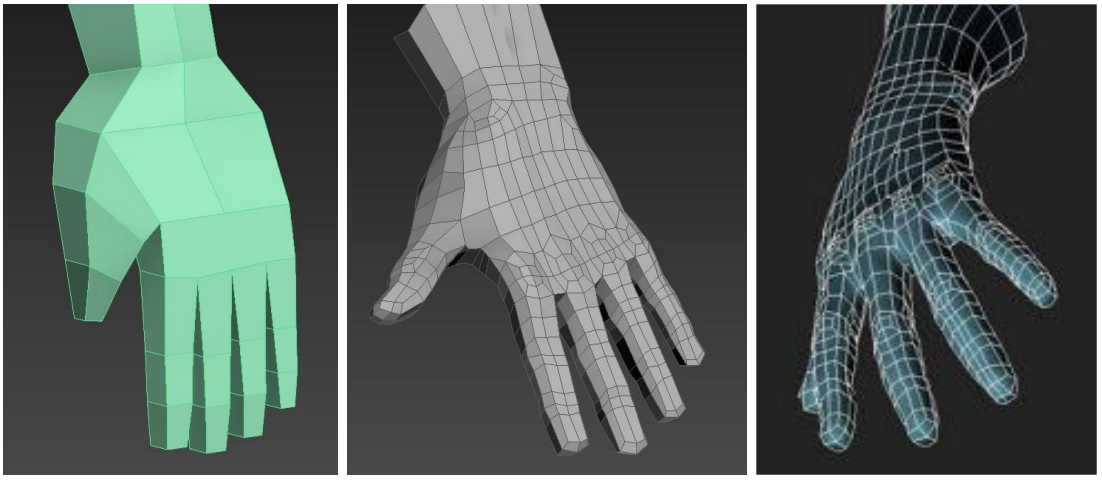
- **Skull modelling process**

The first and most complex part to make was the skull model. It was very time consuming to try to produce a highly detailed skull. As shown in the pictures below I tried to keep it simple at the first stage of the process, I used half of a sphere for the upper part of the crane and an extruded rectangle for the jaw. The whole model uses quadrilateral polygons as the main modelling shape and triangles in some particular cases, this modelling method was suggested by the 3D modelling teacher. As can be seen, the model is highly detailed and the teeth were made one by one instead of relying this task to a texture. The final result was smoothed to add even more detail.



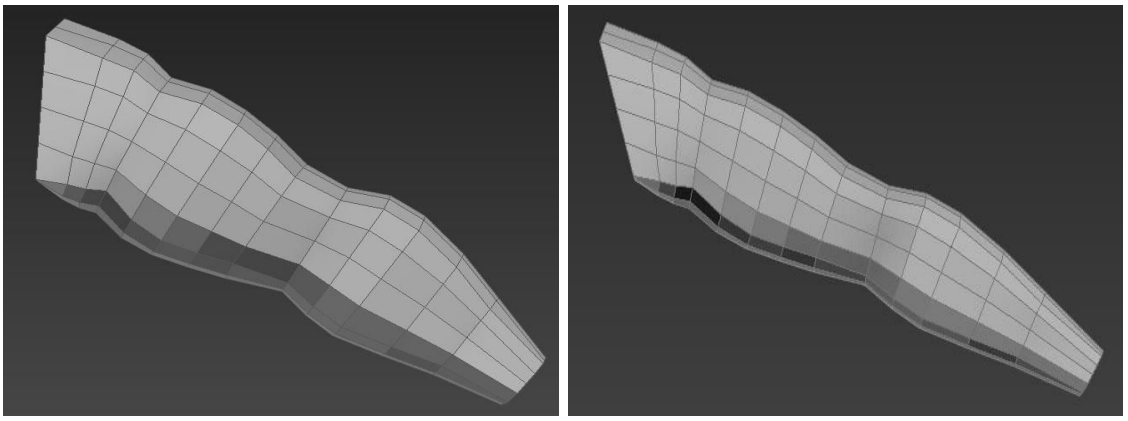
- **Hand modelling process**

At an early stage of development I thought about modeling a simple hand but I changed my mind in order to follow the realistic style that the team wanted for the project. Modelling a detailed hand was a pretty hard job. As shown in the pictures, moving from a simple model to a complex one really required many working hours because every aspect matters to keep the model real: the knuckles, the nails and the thumb were the parts that presented the greatest difficulty. As for the skull, the final result was smoothed.



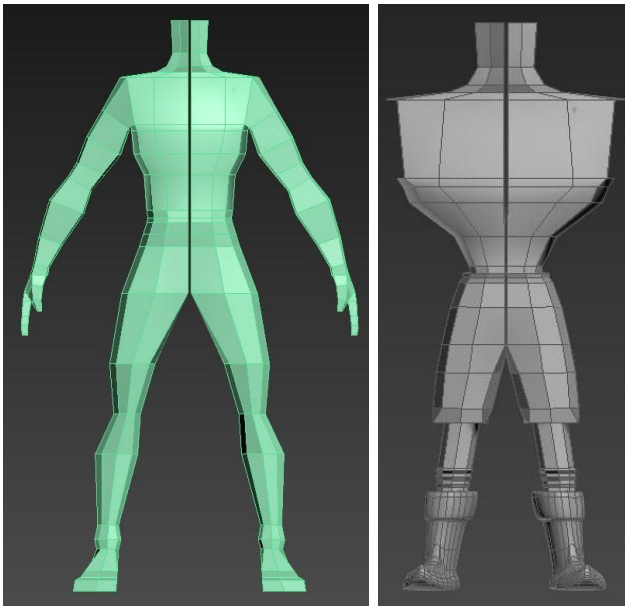
- **Arm modelling process**

  For the arms I used the same process that I had employed to create the rest; start from a simple shape, apply complexity and finally smooth it. The arm model has two variations, a regular one and a strong one, so the team could vote the one that fit the game better. The first version of the character used the strong one, but then we decided to change it so the addition of the armour did not cause problems because of the thickness of the arms. Also, the slims arms were easier to work with while animating the character.



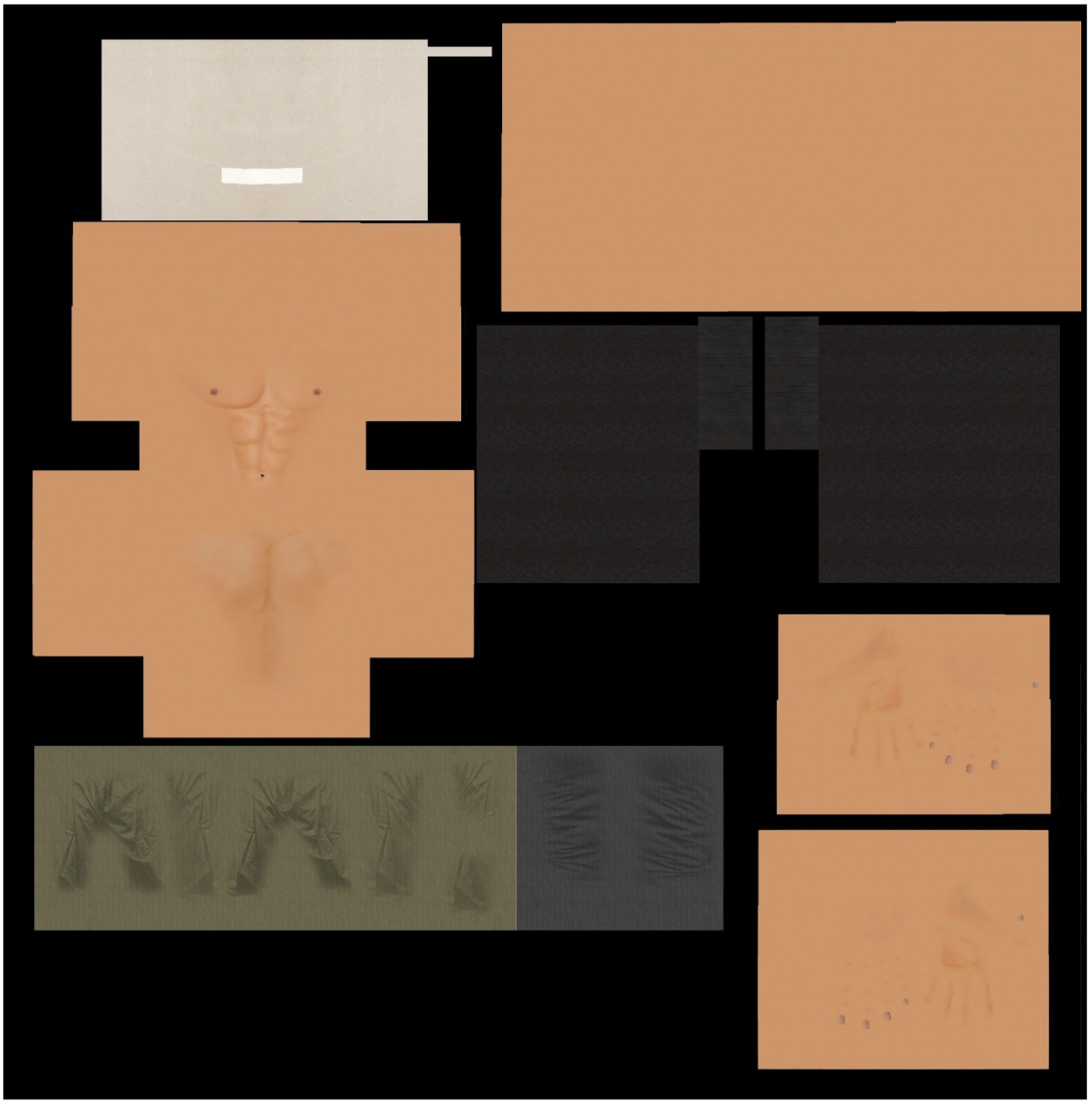
- **Body modelling process**

  For the rest of the body I used the same explained technique one again. Even though I could have separated this model into three different ones, torso, legs and boots, I decided to keep it together because it didn't bother me at all, moreover, in this case was handier to work like this because this way I didn't have to join this meshes.



- **Texturing**

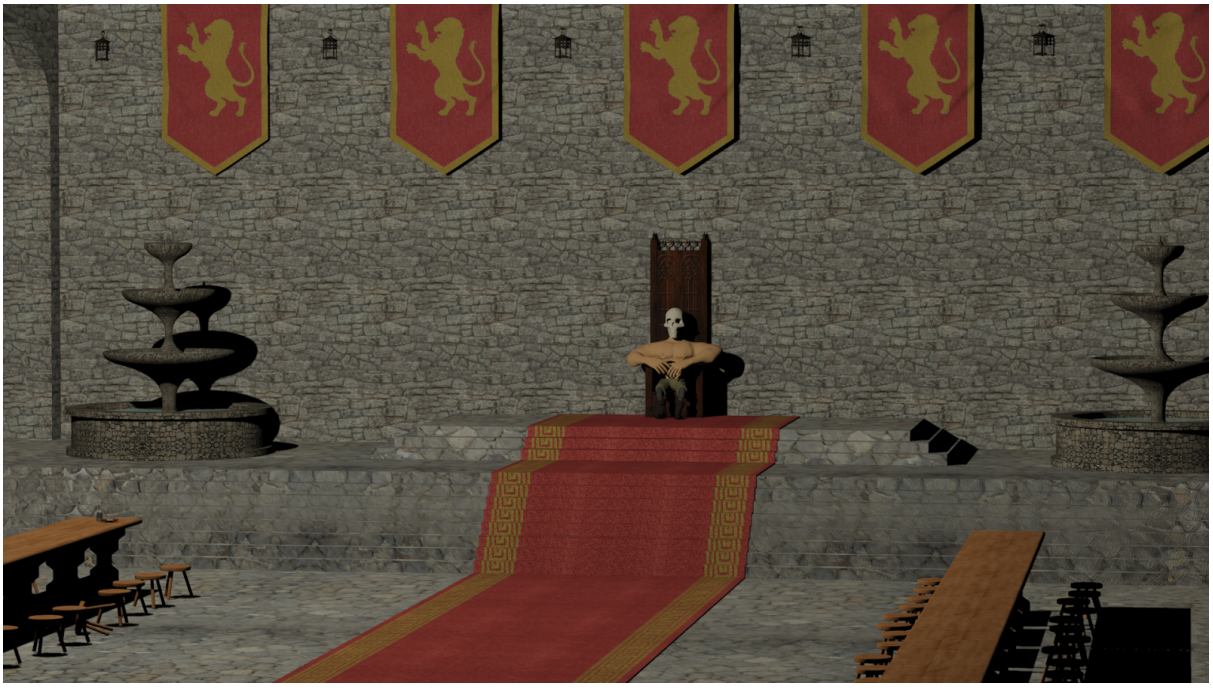
The picture on the right shows the 2D image that supposes the texture for the whole character model. The white part on the top left corner is obviously for the skull. The part below is used for the chest and the back, and then the pants. On the right side we can found the arms texture on top, followed by the boots texture and the hands. The picture shows a high quality textures with many details as the torso muscles, the pants wrinkles or even the nails.



In conclusion, the pictures below show some renders of the first version of the character into a placeholder scene and also some screenshots of the final versión both in the game and in the modelling program.

In the following picture we can see the final boss during his walking animation. Despite of the fact that the animations were made for the first version of the character, they work perfectly on the final version because the changes were very little ones.

The next render displays the boss character seated on his throne. The character also has a getting up animation.



In this last two pictures we can see the final version of the final boss character (left picture). The noticeable changes are that I made it slimmer, took the neck of, changed the boots and put an armour on it. The other picture (right) shows the character in game fighting against the player.

Chapter 6:

Results

This chapter is designed to discuss about the objectives and expectations established at the beginning of the project in the technical proposal. As this report is divided into two different parts, the team project and my individual contribution, this chapter is going to be divided as well. For both parts it is going to be discussed whether the objectives and the expectations were fulfilled or not.

Team Results

- ### Fulfillment of objectives

    - *Create a game based in the action / adventure genre with medieval style that differentiates itself from the rest by its unique qualities.*

      The final thoughts about this objective is that the team archived it. At the end of the development period a fantasy medieval action / adventure game was developed showing all the characteristics of the genre but also a distinguishing difficulty increasing system based on leveling down.

    - *Perform a job close to a commercial level.*

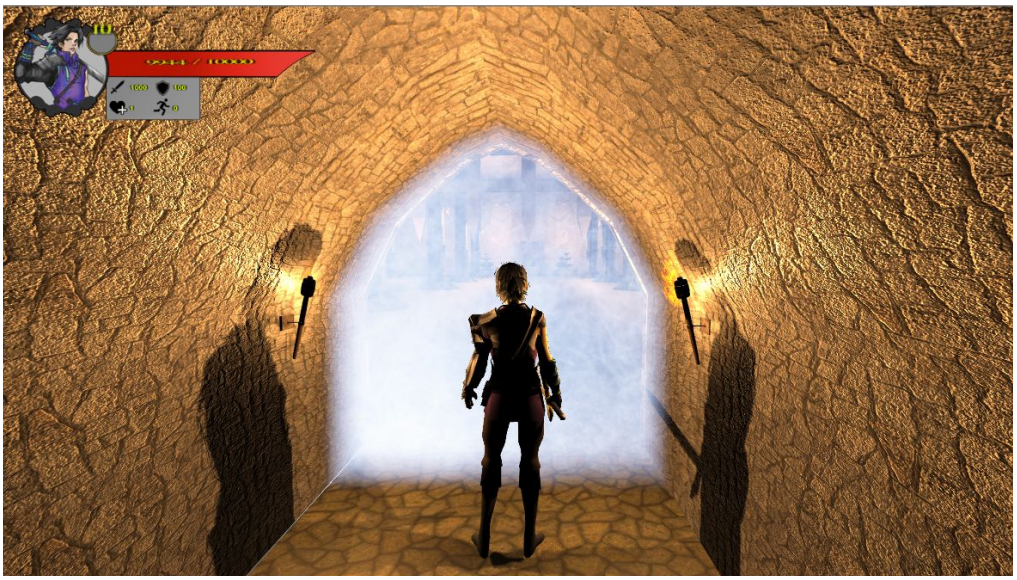
      The final aesthetics and mechanics of the final product video game a lot to a commercial product developed by a real game studio. The used style looks clean and the game works perfectly even though only one testing session was organised at the halfway point of the development and not at the end. Some screenshots of the final product are displayed below:

- *Experience the group work the same way as in a professional level.*

  This was maybe the most successfully achieved objective. Everyone had to experience the real professional experience from the very beginning of the project. Starting from the process to form the work teams: some students were named CEOs and had to explain their ideas, then the rest of the class had to apply for a “job”. The CEOs selected their art and programming team based on the interviews made to the applicants. From that point to the end, the experience continued been as in the real industry, we had to work together with our teammates and the CEO of each group was the one who joined the artwork with the code. We also had testing sessions and weekly meetings.

- **Fulfillment of expected results**
    - *Achieve the development of a brief game of high quality at a visual and mechanical level.*

      This expectation was accomplished successfully, the final product displays some outstanding aesthetics and great mechanics. The pictures below are a good proof of that. The style used the design the levels turned out very realistic and there are not bugs or issues that make the game unplayable or uncomfortable to play. The game runs smoothly and look great.



- *Learn the workings of a real working group of the video game industry.*

  As mentioned before the whole project tried to mimic the working method of the real video game developing industry, where the game studios are divided into teams and there are some people in charge of the work linking to develop the final product. As a team we were able to fulfill this expectation, we learned how to work together as a team, being able to help each other and work in a way that everything were easier to understand for the rest of the members of the team. Also, we learned to follow orders from the person in charge, in this case the CEO of the project.

Individual Results

- ### Fulfillment of objectives

  - *Design, model and texturise quality models for the game scenario.*

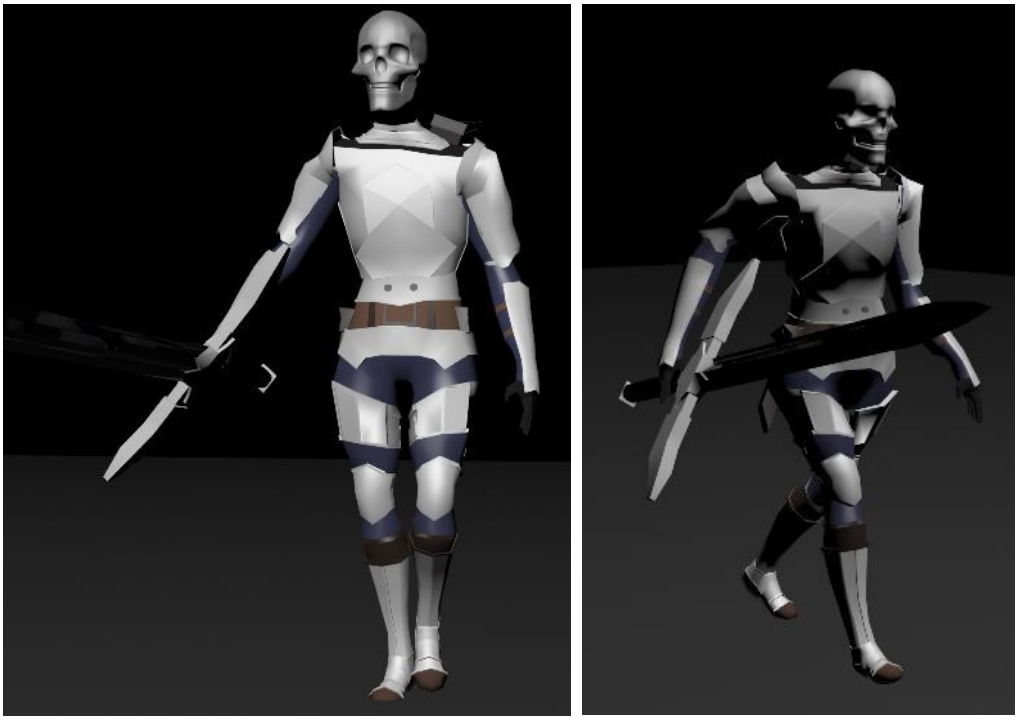
    This was the main objective of my personal contribution to the project and the result is a satisfactory success. The set of models developed not only was a really high quality one, it also was ready far before deadline, which means that I could make another set of just decorative assets to make the game experience even more immersive. Some of the models of the set are shown in the picture below.

  - *Program a profesional third person camera controller.*

    The camera controller developed successfully exceeds expectations. As it is a third person based game, the camera follows the main character and rotates around him but it is also able to move towards the player in order to avoid obstacles and is also clamped so it can not turn upside down by rotating vertically too much. The pictures below show the obstacle avoiding feature (The camera approaches to the player to avoid crossing the wall):

- *Design and program the AI of a character who acts as final boss of the level.*

  This objective was also accomplished by fully creating a final boss character. I made the whole artistic part, which involves modelling, texturing and animating the character, and also the programming part, that means the movement, attack and health scripts. The character turned out quite well and fit the rest of the game perfectly. There are some pictures of the final boss below and the whole scripts can be found in the chapter four of this report.



- **Fulfillment of expected results**
  - *Develop a set of models with their respective professional quality and reusable textures.*

    The models quality overcame the expectations, the textured also turned out great and the assets look very realistic and fit the scene pretty well. Pictures of some of the models can be found above in the individual objectives section, in the chapter five and also below this lines.

  - *Develop the model of a character and his high quality textures and animations.*

    The final boss character was created under a high standard of modelling quality and the textures and animations applied were developed paying attention to the smallest details to make the character look as real as possible. The final result can be found in the previous page.

  - *Program an artificial intelligence and a third person camera controller.*

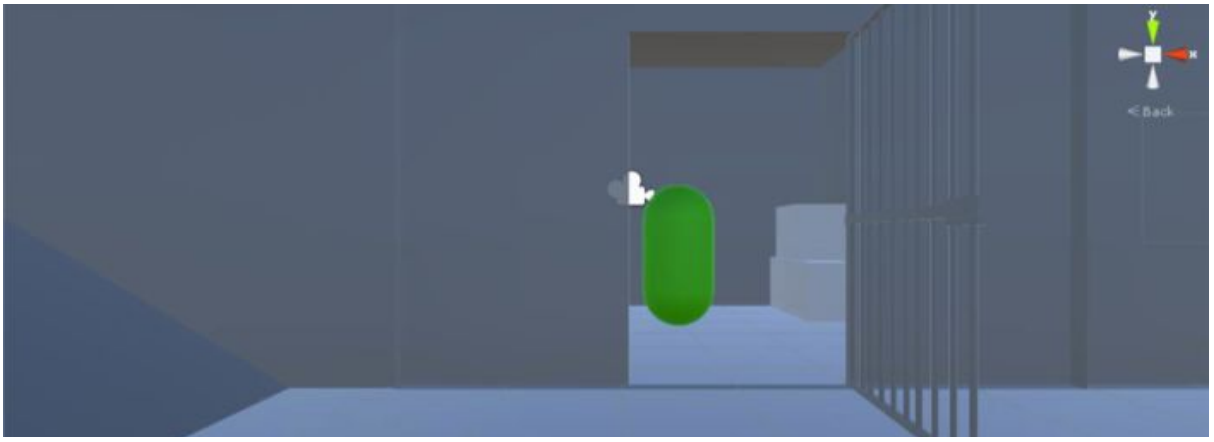
    Both the character's AI and the camera controller were developed and included in the final game causing no problems of any kind. They fit the game and added a nice touch to it, specially the camera controller which excelled the first expectations.

Chapter 7:
Testing & Evaluation

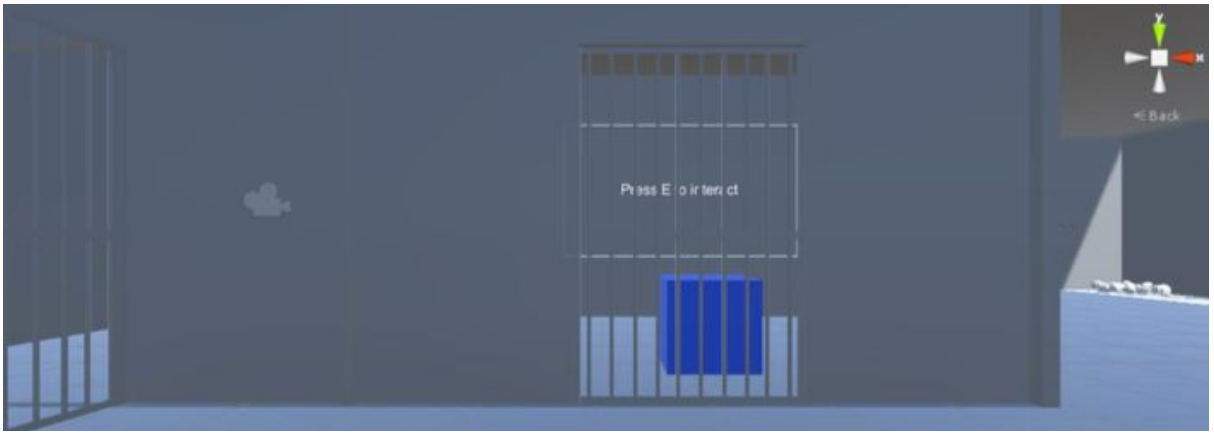
Halfway through the development of the project a test level was released to test the basic mechanics of the game such as the movement system, the dialog system, the inventory and more. This test level was tested by a group of people of the class and their feedback was gathered up in order to improve the core aspects of the game and collect new ideas for the second half of the project development. Part of this feedback is shown later in this chapter.

Test Level:

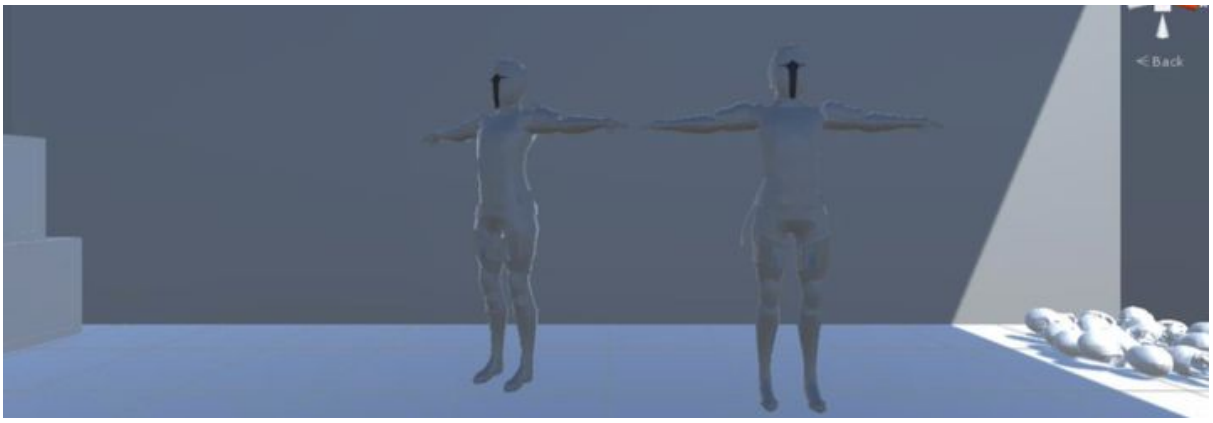
Our grey box level consisted of a basic model of the first room you'll explore in the game along with some basic interior and character models.



The green pill shape in the image above is a placeholder for the player surrounded in the basic cell they'll be trapped inside of at the beginning of the game.



The blue box in the image above is a placeholder for the old man NPC with "Press E to interact" text to prompt the player to engage in character interaction.



The image above shows the guard models put into place to give an idea of what our character models will look like although this models were not yet animated.

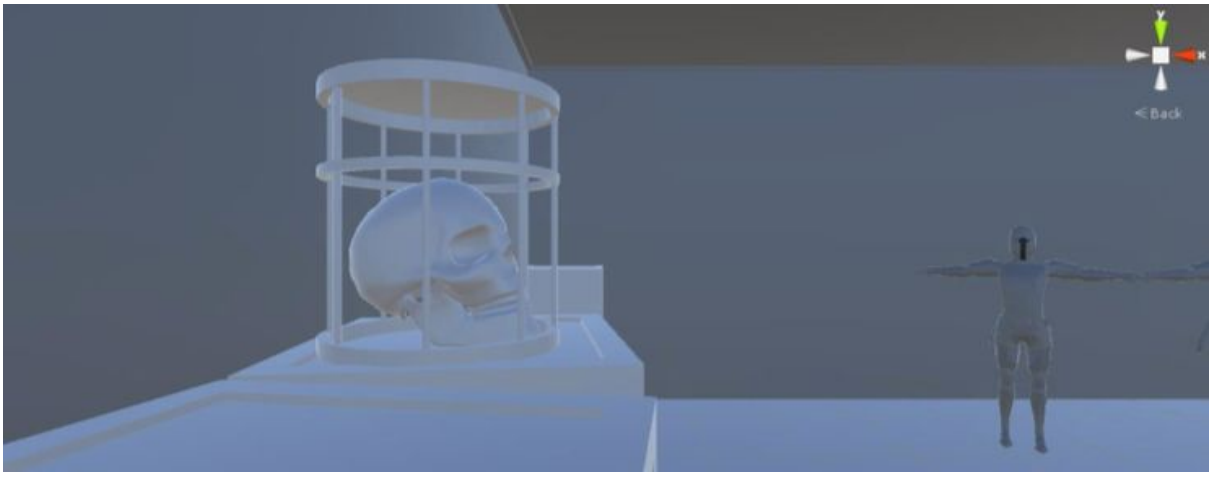
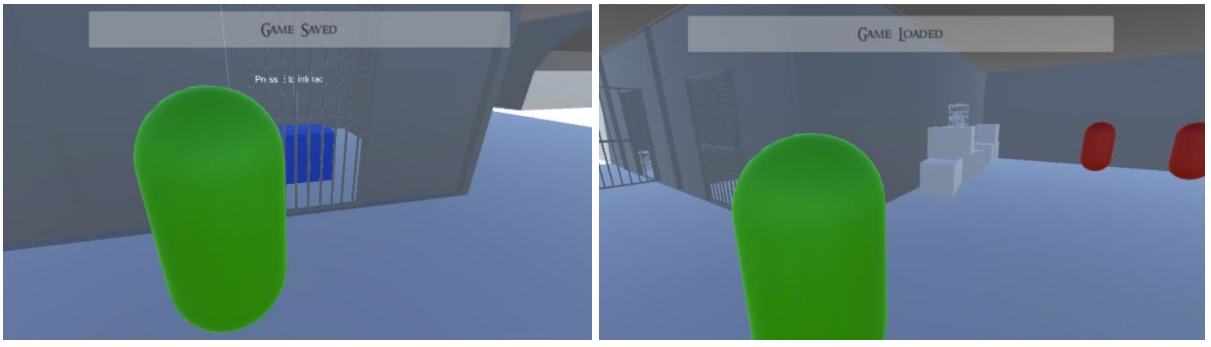


Image above also shows the start of some interior models we have such as boxes, cages and skulls to try and signify to the player that this is a dungeon and a dangerous place to be.

The basic mechanics that were functional in the game was the inventory system, save and load, walking and character interaction. The image below shows an example of the last one.

The image above shows an example of the inventory system.



The two images above show an example of the save / load system.

Player Feedback

Below are two examples of data collected from the testing session which allowed us to see what is currently working and not working for our game. Along with the tester comments, the changes that they inspired are shown (the texts in *italic* style are the changes):

Tester name: Joel Lawton

Age: 20

Favourite genre: Action

Favourite game: The legend of Zelda: Breath of the Wild

Hours per week: 10

Thoughts on the environment: “Overall good, textures need to a bit of touching up, some parts can be seen through. Maybe add some particle effects.” [5/10]

- *Textures were redone after the testing session to become more realistic.*

Thoughts on the characters: “Really nice models, nice detail, needs texturing.” [8/10]

Thoughts on UI: “Some things were not needed like weapon etc. the health bar was really nice, maybe change the health units.” [5/10]

- *A sword was added*
- *The health units were changed.*

Thoughts on menu(s): “Quite nice main menu, add options menu, background art and effects is nice. Inventory is good. Maybe edit the text so it is not so squished.” [7/10]

- *Texts font changed to look less squished.*

Thoughts on mechanics: “Combat needs some work, inventory and pick up worked very well, the AI was a bit too basic. Maybe add some sort of puzzle?” [6/10]

- *Combat system redone, adding smarter IA and realistic animations.*
- *Inventory system upgraded.*
- *Puzzle room added.*

Overall thoughts: “Good working progress.” [6/10]

Tester name: Morgan Clark

Age: 20

Favourite genre: RPG

Favourite game: Monster Hunter 3 Ultimate

Hours per week: 12

Thoughts on the environment: “Environment and props were really good.” [8/10]

Thoughts on the characters: “They’re really good.” [9/10]

Thoughts on UI: “Some aspects are a bit hard to understand.” [4/10]

- *Explanatory texts added to guide the player throughout the game.*

Thoughts on menu(s): “Nice menu, the inventory system is great.” [7/10]

Thoughts on mechanics: “Everything was pretty sound.” [7/10]

Overall thoughts: “Really good so far, more work can be done in some areas.” [8/10]

Chapter 8:
Deviations from the project

At the beginning of the project, apart from the objectives and expected results, a task chart was created with the estimated duration of each task. This chart is the following and is also present in the technical proposal in chapter one.

Task	Hours	1	2	3	4	5	6	7	8	9	10	11	12
Technical proposal	3	3											
Artistic Design	20	15	5										
Modeling	120		20	20	20	20	20	20					
Texturing	50							10	20	20			
Animation	20									10	10		
Programming	35					10					15	10	
Documentation	20											5	15

Although the objectives and expected results were fulfilled, as seen in the last chapter, the time distribution varied slightly. The following chart shows the real time spent on each task, approximately. The tasks that took more time than expected are highlighted in red, and the one that took less, in green:

Task	Hours	1	2	3	4	5	6	7	8	9	10	11	12
Technical proposal	5	5											
Artistic Design	15	10				5							
Modeling	120		20	20	20	20	20	10	10				
Texturing	50						10	10	10	20			
Animation	25									10	10	5	
Programming	35					10					15	10	
Documentation	35											5	35

The first notable thing about the table is that the technical proposal took about two hours more than the expected three hours. I might be because of the time spent on translating but this small change made no difference anyways.

The artistic design took about five hours less than expected, this is because the assets design was pretty easy because it is based in real objects. Moreover, the character design was also less time consuming than expected because of the strong use of references.

The modelling and texturing part took approximately the expected time but at the halfway point of the project I decided to split some of the weeks into two to relief some work pressure and mix two kind of tasks the same week because doing the same thing all the time can make the quality of the product decrease due to I can get tired.

The animation took some more hours than expected and an extra week had to be added to the schedule. The animating process was hard in order to create quality content similar to real life movement.

The programming task was the one that adjusted the best to the schedule. It really took about the expected time and could be done within the time specified in the task chart.

The task most unexpectedly different from the initial schedule was the documentation part. Gathering all the needed documents took a lot of time but the most task consuming part was the final report writing.

Chapter 9:
Conclusions

The content of this chapter is based in my personal opinion and experiences. It includes a brief summary of the project development, what I learned from the making of the game and all the documents related to it, and the fulfillment or not of the objectives established at the beginning.

Summary of the project

In the first place I want to make a brief overview of the whole project. It is a fantasy medieval action / adventure video game in third person and with realistic aesthetics. The plot describes the story of a character who founds himself in a prison and unable to remember anything. The objective of the game consist in escaping from this place but in order to archive this goal the player have to fight against many guards and a final boss while solving puzzles that allow him to advance.

Personal experience

This project has given me the chance to improve my artistic skills a lot. I have spent a lot of time modelling in 3D, creating and applying textures and designing animations. I also have learned a lot about programming.

Also I was able to live a real work experience while doing this project and I have discovered that I like it and I would like to work as a designer or three dimensional modeller.

I have found some subjects studies during the degree very helpful in order to develop this project, specially the following ones:

- 3D design (UJI): As my job was mainly 3D modelling, this was the most important subject to take reference from. 3D design taught me the basics of 3DS Max, which is nowadays my favourite program to work with, it is simple to use, very intuitive and allows the user to produce outstanding results.
- Character design and animation (UJI): As modelling and animating a character was one of my tasks, this subject that taught me the basics of character modelling and biped animation was really handy.
- Programming II (UJI): This is the subject that establishes the base of C# programming in the degree, so it was the starting point for all my programming tasks of the project.
- Artificial Intelligence (UJI): In AI I learned how to implement the whole decision making system of a character, from where to move to who to attack and how its health works, and so on.

- Games Design Prototyping (UoS): This was the most important module in order to make the project because it supposed the project itself. My job for this subject composes the individual contribution shown in this report. Thank to this module I learned the basics of working within a group divided into different team, following the orders of another member of the group and also applying for the “job”.

- Digital Character Art (UoS): Last, this module was similar to the character design and animation subject mentioned before but more advances. I learned how to develop a high quality character model and to apply textures to it. Also, I learned how to make quality animation that resemble to real life human movement.

Fulfillment of objectives

In conclusion, I would like to talk about the objectives established at the beginning of the project, both for the team and for myself. I already explained this objectives and expectations fulfilment in the chapter six, where I describe the results obtained.

In this case, I just want to emphasize in how successfully the objectives were achieved, as a team we were able to develop a high quality fully functional video game with a very visual aesthetics. As for my personal objectives I was able to accomplish everything I proposed, I developed a high quality set of environment assets and I was also able to program a camera controller and a final boss character AI scripts.

In a nutshell, this has been a very satisfactory experience where I have been able to learn a lot about the video game developing industry and I have highly improved my skills.

References

[1] The Lord of the Rings:
https://en.wikipedia.org/wiki/The_Lord_of_the_Rings_(film_series)

[2] Game of Thrones:
http://en.wikipedia.org/wiki/Game_of_Thrones

[3] The Legend of Zelda:
http://zelda.com/

[4] The Elder Scrolls:
http://es.wikipedia.org/wiki/The_Elder_Scrolls

[5] MediEvil:
https://es.wikipedia.org/wiki/MediEvil

[6] World of Warcraft:
https://worldofwarcraft.com/es-es/

[7] Diablo:
https://eu.battle.net/d3/es/

[8] God of War:
https://godofwar.playstation.com/

[9] Prototype:
https://en.wikipedia.org/wiki/Prototype_(video_game)

[10] Shadow of the Colossus:
https://es.wikipedia.org/wiki/Shadow_of_the_Colossus

[11] Merlin:
http://www.bbc.co.uk/programmes/b00mjlxv

[12] The Legend of King Arthur:
http://en.wikipedia.org/wiki/King_Arthur

[13] The Witcher:
http://thewitcher.com/en/witcher3

[14] Unity 3D:
https://unity3d.com/

[15] Visual Studio:
https://www.visualstudio.com/

[16] Teamviewer:
https://www.teamviewer.com/

[17] Slack:
https://slack.com/

[18] Adobe Photoshop:
http://www.adobe.com/es/products/photoshop.html

[19] Autodesk 3DS Max
https://www.autodesk.com/products/3ds-max/overview

[20] Adobe Audition
http://www.adobe.com/es/products/audition.html

Appendices

Appendix 1: Games Design Prototyping module information

University of Salford MANCHESTER		School of Arts & Media **Module Information Summary** (MIS)[1]
		Module title: Games Design Prototyping
Academic Yr: 2016_2017	**Level:** 5	**Programmes on which offered[2]:** BSc(Hons) Computer And Video Games
Credits: 20	**Duration[3]:** 12 Weeks	

Staff:

Module Leader	**Teaching Staff**		**Support Staff**
Juan Hiriart	Juan Hiriart / Mick Lockwood		Rod Martin
Internal verification complete?[4]	yes	**Internal verifier[5]**	Roy Chilvers
External verification complete?[6]	yes	**External verifier[7]**	Dr Christos Gatzidis

Risk Assessment[8]:

Date completed	**Completed by**	**Special instructions**
15/02/2017	Juan Hiriart / Mick Lockwood	None

Rational

The work will be undertaken as a group. The groups will be made up of 5/6 individuals and established by tutors. Any individual who can't find a group to work in will be placed after consultation with your Lecturer and Year Tutor.

Working within a group it is important that each and every member contributes, a proportion of the work will be undertaken as an individual and a proportion will undertake as a collective. A team contract will be negotiated at the start of the project and basic roles will be assigned, attendance will be kept, individuals will be monitored, and a log will be submitted with the team submission so that we have transparency. In the unlikely event that one individual fails to contribute, adjustments can be made accordingly.

It is expected that each group will develop, from conception to completion, a full 3D game prototype using the UNITY game engine and supporting 2D - 3D authoring software. As a requisite for this semester, the game now requires to implement 3D skinned meshes fully animated, and controlled using the engine's in-built animation system (Mecanim). This will give you the opportunity to experiment and acquire valuable knowledge

[1] Please complete this form in reference to the approved Module Specification form (MS1) – go to the online repository for the latest version: https://collaboration.salford.ac.uk/ops/pmsc/SitePages/Home.aspx. Guidance notes are provided for staff benefit – these can be removed as you finalise the document for student use.
[2] Please indicate the programmes the module forms a part of.
[3] Duration of the module should not exceed 12 weeks for modules that operate within a semester. 'Long, thin' modules should not exceed week 12 of the second semester in which they operate.
[4] Please indicate 'yes' once internal verification is complete. The standard verification template must be completed in addition to this form.
[5] Please provide the name of the internal verifier who should be an academic who is independent of the module.
[6] Please indicate 'yes' unless the module is not subject to external verification where NA should be indicated. (Refer to the policy 'Assessment & Feedback for Taught Awards' for further guidance: http://www.governance.salford.ac.uk/cms/resources/uploads/File/AQA/Assessment_and_Feedback_for_Taught_Awards.pdf.) The standard verification template must be completed in addition to this form.
[7] Please provide the name of the external verifier who should be the relevant External Examiner.
[8] Please complete in reference to the School's Health & Safety policies and procedures.

and experience with state of the art animation technologies, such as hierarchical state machines, procedural animation systems and mesh deformation algorithms. We are aware that the introduction of these technologies will considerably raise the level of complexity of your project, so especial tutorials and workshops will be set during the semester in order to help you acquire a firm grasp of all the technologies involved.

The game will be iteratively developed throughout the semester, with two formal testing sessions projected at its alpha and beta state. Along with the prototype, each group will also have to hand in a set of communication materials intended to inform the design ideas, concepts and development process carried during the semester. All these materials are covered in detail in the deliverables section.

Module aims[9]:

Aims of Module:

a) To facilitate a synthesis of design practice and digital animation production within its contextual framework
b) To provide an opportunity for students to take responsibility for the negotiation, management and implementation of a small scale digital game prototype.
c) To encourage students to bring together skills, knowledge and understanding from different areas of the course in a collaborative effort.
d) To develop self-management, independent learning, decision making and autonomy in the implementation of a digital game prototype.
e) To enable students to realise innovative aspects of traditional and digital game production.
f) To encourage both creative and technical skill within all aspects of project implementation.
g) To formulate innovative and original approaches in creative practice and research.
h) To develop the ability to utilise emerging game technologies.

Module intended learning outcomes[10]:

Knowledge and Understanding
On completion of this module the student should be able to:

1. Select appropriate technologies towards the delivering of high standard of product suitable for distribution and exhibition
2. Report on the game prototype production process in a written form through a self reflective development diary
3. Describe and discuss the relevant stages involved in digital game prototype production.

Intellectual Skills

4. Formulate the project as a live product development scenario
5. Demonstrate innovative design & production & technical ability and project management to complete the project.
6. Demonstrate an in-depth understanding and knowledge of their chosen area within development and production practice.
7. Synthesise concepts & industry practice to produce a functional prototype that demonstrates original work.
8. Present a game prototype project in oral forms and compare the final outcome in comparison to industry and 'indie' practice.

Transferable/Key Skills and other attributes

- Decision making
- Problem solving
- Planning

[9] Ensure information provided here matches the approved detail as it is outlined in the relevant MS1.
[10] Ensure information provided here matches the approved detail as it is outlined in the relevant MS1.

- Oral communication
- Written communication
- Adapting
- Researching information
- Flexibility
- Time management

Module schedule or scheme of work (content, dates, activities, interim deadlines, lecture programme, etc.):

The module will consist of formal lectures, workshop sessions, seminar and individual and group tutorials. Within the module you will work within a small group to generate ideas, develop concepts, build prototype and test and refine your offering. The development of the game will be documented in a lightweight design document developed in an iterative process.

Sprint 1	Week 1	30.01.2017	• Module's overview • Initial game concept ideation (1-2 members)
	Week 2	06.02.2017	• Presentation of initial game concepts • Group formation
Sprint 2	Week 3	13.02.2017	• The third person controller. Aesthetic and technical aspects
	Week 4	20.02.2017	**Milestone 1** • Submission (formative assessment): One-page Game Design Document, Moodboards and concept art, and game sketching. • Pitch presentation of game idea (formative assessment).
Sprint 3	Week 5	27.02.2017	• Animation controllers (Mecanim) • Project follow-up
	Week 6	06.03.2017	• Game narrative theory • Developing narrative components: conversation trees. • Sprint review
Sprint 4	Week 7	13.03.2017	• Advanced user interfaces (Inventories) • Saving game data. • Project follow-up
	Week 8	20.03.2017	**Milestone 2** • Internal testing session (formative assessment): Grey box prototype, character/story development. • Sprint review
Sprint 5	Week 9	27.03.2017	• Project follow-up • Individual and group tutorials
	Easter vacation		

	Week 10	24.04.2017	**Milestone 3** • External play-testing (formative asessment): asset integration (80%), level and interface design. • Sprint review
Sprint 6	Week 11	01.05.2017	• Project follow-up • Individual and group tutorials
	Week 12	08.05.2017	• Individual and group tutorials

Assignment details (project outline, essay questions, submission information, etc.):

Component 1: Group Project (80%)

Game Prototype (40%)

A final playable game prototype will be submitted at the end of the semester. This component requires to be submitted in a digital form, consisting in a compiled version of the game (Windows and Mac), and a separate folder with the full game project, organised in an appropriate folder structure. Please make sure that your disk is readable, and contains the required compiled versions of your game.

The game prototype needs to demonstrate your design ideas in terms of gameplay, mechanics, aesthetics, and narrative. All conceivable play situations need to be covered by appropriate rules, and the game is well balanced, giving evidence of a systematic play-testing process.

All the systems required to do so need to be implemented with no major bugs remaining. Also, game aesthetics should be developed to a polished level across all in-game assets: game graphics, textures, sounds, interface elements.

At this stage your game requires to demonstrate the following components:

- Advanced Interface implementation (Navigable screens, HUDs, etc.)
- Camera controllers
- Character animation controllers
- AI agents (FSM, Goal-driven behaviors, Pathfinding, etc.)
- Persistent data (SharedObject, XML, binary, etc)
- Organic modelling and texturing, Rigging, Animation
- Level design, progression mechanisms

Game Design Document (30%)

A Game Design Document (GDD) should be submitted in a printed and digital format. The document should be able to explain in detail your game proposal while also giving a thorough account of your design process. The document requires to have a clear title, group name, group members, date, module code, and module tutors.

In order to communicate your ideas as clearly as possible, consider to present as much information as possible in a visual form. Although not a strict rule, your document should be no longer than 1000 words and at least 60% of the total extent of the document should be displayed visually, through a combination of pictures, drawings and schemes. All visuals should have captions or be properly referenced in the main text.

Your document requires to give evidence of the design process followed during the semester, presenting the design decisions and changes from initial design ideas. For this purpose, introduce updates instead of

replacing the original information. Updates should be clearly identified from the initial text by brackets containing the date in which the change was introduced.

All external sources need to be properly referenced following the APA Harvard model (download and use the following guide:
http://www.salford.ac.uk/library/help/user-guides/general/Bibliographic-Citations-APA-6th-Harvard-Feb14.pdf)

The document will be structured as follows:

1.0 X Statement.
Write a concise statement that defines both core gameplay and how your game will be positioned in the market. This statement will guide both the game development and marketing efforts (approx. 150 words)

2.0 Game Concept.
Make a description of your game in terms of genre and theme. Present some existent games as references. Include a series of bullet points with an overview of your game idea.

3.0 Conceptual topic
Your game is designed to explore a games design topic of interest. This section offers an overview of the selected topic, relevant literature and conclusions.

4.0 Aesthetics.
Describe how your game will look, sound, and feel (also tastes and smells, if necessary). This section should cover everything, from soundtrack and character design to packaging and cover art. Also, include here your references (from internet and real world
sources), initial concepts, storyboards and visual records of Cave Expo.

5.0 Mechanics.
Define briefly all the systems that define how the player and the game interact. Include player interaction patterns, objectives, rules, resources, and boundaries..

6.0 Story.
Write down your back-story and the fictional world in which the game is based. Describe also the narrative components that will be used to communicate the story to the player (Cards, Graphics, Counters, etc.)

7.0 Technology.
This section describes the underlying technology that will make your game work. Please take into consideration the ways in which the chosen technology will affect your design decisions and plan ahead feasible ways to overcome the potential limitations that it presents. Include in this section your programming scope and additional requirements.

8.0 Interface.
Describe the interface menus, HUDs, and GUI elements that your game will use to support gameplay, narrative, social components, etc. This section should be presented graphically, through wireframes, story-boards and paper-based low fidelity prototypes.

9.0 Sandbox plan.
Visual schematic map defining the spatial positioning of gameplay elements (enemies, pick-ups, etc.) Provide brief explanations supported by tight vector graphics.

10.0 Test level (Walkthrough)
Write a hypothetical player path describing progression mechanism, typical or recurrent playing situations, puzzles, loosing conditions, level structure, etc.

Project Documentation (10%)

Each group will develop their project tightly following an AGILE methodology. As a result it is expected that each group will produce a substantial amount of management documentation, which need to be submitted at the end of the semester. This documentation includes:

- Sprint plan and analysis. Each group will submit a sprint report at the end of each sprint. The report can be downloaded from the Blackboard under the section "Resources". The report provides a summary of the tasks completed from the sprint backlog, and each one of the completed tasks should be linked to the individual Tumblr account of the task responsible.
- Progression of War Room. Pictures of schemes, sketches, concept art and any material displayed in the assigned wall space during the project's development.
- Internal testing session raw data.

Pitch Presentation (20%)

The presentation will be conducted within the CVG designated room and will be supported by a keynote presentation, visual elements taken from your War Room Expo, project documentation, and game prototype.

Each group will have 20min of time to deliver their presentations, followed by a Q&A session of 10min. It is important for each group to design and prepare your presentation so you do not exceed the allocated time. Although this is certainly important and will influence our final mark, you wont be penalised for taking some extra seconds in finishing your presentation (e.g. finishing your sentences in the last slide of your presentation). If however your presentation exceeds this acceptable extra time, this could have an effect in your final mark.

Each group will decide which member(s) will deliver the presentation. You can select only one member or decide for all the group members to speak. All group members however need to be prepared to actively participate and answer during the following Q&A session.
You are completely entitled to use communicational gestures in support of your speech. Gestures and body language can greatly enhance your presentation and help you to persuade your audience, but they need to be used wisely and naturally.

The definition of the structure to follow in your presentation is part of the assignment, and will depend on the ideas that you consider more important to communicate about your project, and the aspects that you wish to emphasise. As a suggestion, do not just repeat the information that you are already delivering in your project documentation, but complement that with further insights and deeper reflection.

Component 2: Individual Contribution Report (20%)

- Physically printed record of your weekly contribution to the group project, supported by access to your Tumblr account.
- Physically printed copy of Peer Assessment Form (available from Blackboard).
- Individual Contribution Interview

* Further guidance on all deliverables will be offered during class sessions.

Assessment criteria for module[11]:

1. Creativity
2. Critical Judgement and Analytical Ability
3. Enquiry and Use of Sources
4. Knowledge and Understanding of Subject Matter
5. Visual Communication
6. Oral Communication
7. Technical Skills

[11] List all assessment criteria to be utilised within the module here and number – cross-reference to 'specific assessment criteria' box.

8. Team Working

Summary of assessed components[12]**:**

Component 1: Group Project (80%)

Game Prototype (40%)

- Digital submission. Please make sure that your disk runs properly, and is correctly labelled with your group's name and members.
- Folder containing the final compiled version of your game for both Mac OSX and PC Windows operative systems. Include versions for other platforms, if relevant (Android, IOS, etc.)
- Organised folders containing the whole game project, with all the assets used in production.

Lightweight Design Document (30%)

- Digital and printed PDF submission of your final document.

Pitch Presentation (20%)

- Digital submission of your keynote presentation (PDF format)

Project Management Documentation (10%)

- Document containing all sprint reports.
- Progression of War Room. Pictures of schemes, sketches, concept art and any material displayed in the assigned wall space during the project's development.
- Internal testing session raw data.

Component 2: Individual Contribution Report (20%)

- Physically printed record of your weekly contribution to the group project, supported by access to your Tumblr account.
- Physically printed copy of Peer Assessment Form (available from Blackboard).
- Individual Contribution Interview

Component description	**Weight**	**Deadline**	**Submission location**[13] (if excepted from e-submission)	**Specific assessment criteria**[14]
Individual Contribution Report	**20%**	19.05.2017	Submitted in Physical Format. MC 3.27 16:00	2,3,4,5
Group Project	**80%**	19.05.2017	Game Prototype. Game Document. Physical Format. MC 3.27 16:00	1,2,3,4,5,6,7,8
Information about feedback:				

[12] Ensure information provided here matches the approved detail as it is outlined in the relevant MS1.
[13] Use the following standard statement here unless the component is formally exempted from e-submission: 'Not applicable – submission is electronic – see assignment details for further information.'
[14] Complete by listing numbers of relevant assessment criteria, in reference to 'assessment criteria for module' box. Refer to footnote 10.

Formative feedback points within the module[15]	20.02.2017, 20.03.2017, 24.04.2017
Date of return of summative feedback[16] **– component 1**	09.06.2017
Date of return of summative feedback[17] **– component 2**	09.06.2017

Academic Misconduct

The University takes a serious view of all acts of academic misconduct. Such acts are considered dishonest and as attempts to gain unfair advantage. Acts of academic misconduct can take many forms. They are likely to fall into one or more of the following categories:

• _Plagiarism - taking the work of another person or source and using it as if it were one's own

• _Self-plagiarism (or double submission) – resubmitting previously submitted work on one or more occasions (without proper acknowledgement) including work submitted for credits at a previous institution
• _Collusion - working with others on tasks that should be carried out on an individual basis
• _Falsifying experimental or other investigative results
• _Taking unauthorised material (including electronic devices) into an examination
• _Contracting another person to produce a piece of assessed work
• _Producing a piece of assessed work for another person
• _Copying from, or communicating with, another examination candidate during an examination
• _Bribery

As a student of the University, you should ensure that you know and understand what is meant by academic misconduct and the requirements of your school for referencing and citation. Your academic tutors will provide you with detailed advice and training about good academic practice (i.e. how to avoid academic misconduct) at the start of the academic year and during modules. In addition, you can seek further guidance about what constitutes good academic practice from your module tutors and on BlackBoard. Study skills information about academic misconduct and referencing are available at these links: http://www.careers.salford.ac.uk/studyskills http://www.salford.ac.uk/__data/assets/pdf_file/0005/55679/bibcitation.pdf

When submitting each assignment, you will be required to complete a declaration confirming that the work submitted is your own, with all sources duly acknowledged. Where assessments are submitted electronically, this declaration is completed online.
If academic misconduct is suspected, the University may use facilities such as the Turnitin originality report as part of their investigations.

You are also recommended, when available, to make use of Turnitin (accessed via Blackboard) prior to submission to check your assignments for any referencing/plagiarism issues. This will not be available for all assessment types.
If you are found guilty of academic misconduct, the University will impose a penalty as described in the Academic Misconduct Procedure, available here:
http://www.salford.ac.uk/university/governance/policies-and-procedures/browse-by-theme/2

[15] Provide information here on formative feedback opportunities for students that are provided within the module.
[16] Please record the date by which feedback will have been returned to students ensuring you adhere to the policy of return of feedback to students within three weeks of submission. Where you have one component only please remove reference to component 2.
[17] As above.